Major Project Report

on

REAL TIME FUZZY CLASSIFICATION OF CEPSTRAL COEFFICIENTS FOR SPEAKER IDENTIFICATION

Submitted in partial fulfilment of the requirements for the degree of

MASTER OF TECHNOLOGY

IN

INFORMATION SYSTEMS

Submitted by

SRISHTI

Roll No. 16/ISY/2K10

Under the guidance of

MS. SEBA SUSAN

(Assistant Professor)

Department of Information Technology

Delhi Technological University, Delhi

2010-2012

CERTIFICATE

This is to certify that the work contained in the thesis titled "**Real Time Fuzzy Classification of Cepstral Coefficients for Speaker Identification**" is an original piece of work which has been carried out by **Srishti** under my supervision. This work has not been submitted elsewhere for a degree.

Ms. Seba Susan

Assistant Professor

Department of Information Technology

Delhi Technological University, Delhi

July 2012

Abstract

Speaker Identification refers to using utterances from a speaker, in order to determine who the speaker is out of a set of known speakers. Speaker Identification is a widely popular but a complex problem. It is difficult to design accurate algorithms capable of extracting salient features and matching them in a robust way. A novel system for speaker identification based on fuzzy logic has been proposed in this research. Mel-frequency cepstral coefficients (MFCC), which are commonly used for voice-based biometric recognition, form the voice features. In this thesis, a fuzzy nearest neighbour classifier is built for text independent speaker identification using MFCC features. The fuzzy classifier is based on the Gaussian membership function. Speaker identification experiments using the fuzzy nearest neighbour classifier have been carried on a well known publicly available voice dataset. The performance of the proposed classifier has been compared against some of the other commonly known techniques used for speaker identification. The results obtained by the different techniques have been compared on a number of parameters like the efficiency, the time complexity and the space complexity of each algorithm. The obtained results are very promising and verify our claim that the proposed scheme gives better performance than the already existing techniques for speaker identification and that it has the potential to attain a reasonable real-time performance.

Acknowledgements

I would like to express my sincere gratitude to my advisor, Ms. Seba Susan for giving me an opportunity to work with her and for her advice, encouragement and constant support. I wish to thank her for extending me the greatest freedom in deciding the direction and scope of my research. It has been both a privilege as well as a rewarding experience working with her.

I would also like to thank all my friends and colleagues here at Delhi Technological University for their valuable comments and suggestions, many of which have been vital to this work. Furthermore, I want to thank the faculty of Department of Information Technology and the staff for all their help.

I would also like to take this opportunity to thank my parents for their love, sacrifice, understanding, encouragement and support. I am forever indebted to them for all that they have given me. Finally, special thanks to my brother for his motivation and help.

Srishti

CONTENTS

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Biometrics

The term Biometrics has been derived from the Greek words bio (life) and metrics (measure). [1] Biometrics is the science that is composed of methods and techniques that enable us to uniquely establish the identity of an individual based on one or more intrinsic physiological or behavioural traits. Biometrics may include modalities like voice, face, fingerprint, hand geometry, retina, iris, palmprint, ear structure, gait, keystroke dynamics, etc., (Figure 1.1)

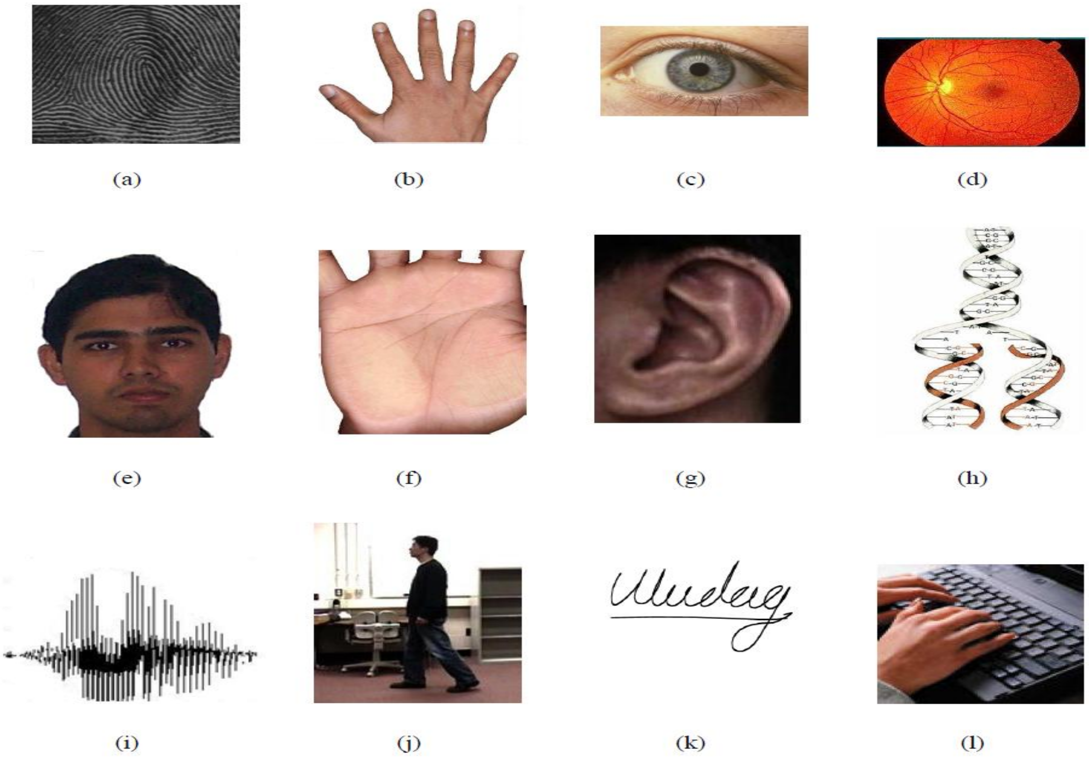


Figure 1.1 : Characteristics that are being used for biometric recognition; (a)Fingerprint; (b) Hand-geometry; (c) Iris; (d) Retina; (e) Face; (f) Palmprint; (g) Ear structure; (h) DNA; (i) Voice; (j) Gait; (k) Signature and (l) Keystroke dynamics. [2]

Biometric recognition has tremendous value in security applications where a reliable identity management system is needed. Examples of such applications include physical access control to a secure facility, e-commerce, access to computer networks and welfare distribution. The primary task in an identity management system is the determination of an individual's identity. Traditional methods of establishing a person’s identity include knowledge-based (e.g., passwords) and token-based (e.g., ID cards) mechanisms. However, these can easily be lost, shared or stolen. Therefore, they are not sufficient for identity verification in the modern day world. Biometrics offers a natural and reliable solution to the problem of identity determination by recognizing individuals based on their characteristics that are inherent to the person.

Biometrics offers several advantages over traditional security measures. These include:
1. Non-repudiation: With token and password based approaches, the perpetrator can always deny committing the crime pleading that his/her password or ID was stolen or compromised even when confronted with an electronic audit trail. There is no way in which his claim can be verified effectively. This is known as the problem of deniability or of ’repudiation’. However, biometrics is indefinitely associated with a user and hence it cannot be lent or stolen making such repudiation infeasible.

2. Accuracy and Security: Password based systems are prone to dictionary and brute force attacks. Furthermore, such systems are as vulnerable as their weakest password. On the other hand, biometric authentication requires the physical presence of the user and therefore cannot be circumvented through a dictionary or brute force style attack. Biometrics have also been shown to possess a higher bit strength compared to password based systems and are therefore inherently secure.

3. Screening: In screening applications, we are interested in preventing the users from assuming multiple identities (e.g. a terrorist using multiple passports to enter a foreign country). This requires that we ensure a person has not already enrolled under another assumed identity before adding his new record into the database. Such screening is not possible using traditional authentication mechanisms and biometrics provides the only available solution [3].

Any human physiological or behavioral characteristic could be a biometrics provided it has the following desirable properties:

(i) Universality, which means that every person should have the characteristic,

(ii) Uniqueness**,** which indicates that no two persons should be the same in terms of the characteristic,

(iii) Permanence, which means that the characteristic should be invariant with time,

(iv) Collectability, which indicates that the characteristic can be measured quantitatively.

In practice, there are some other important requirements:

(v) Performance, which refers to the achievable identification accuracy

(vi) Acceptability, which indicates to what extent people are willing to accept the biometric system, and

(vii) Circumvention, which refers to how easy it is to fool the system by fraudulent techniques.

1.1.1 General Architecture of a Biometric System

In general, biometric verification consists of two stages (Figure 1.2)

(i) Enrolment , and

(ii) Authentication

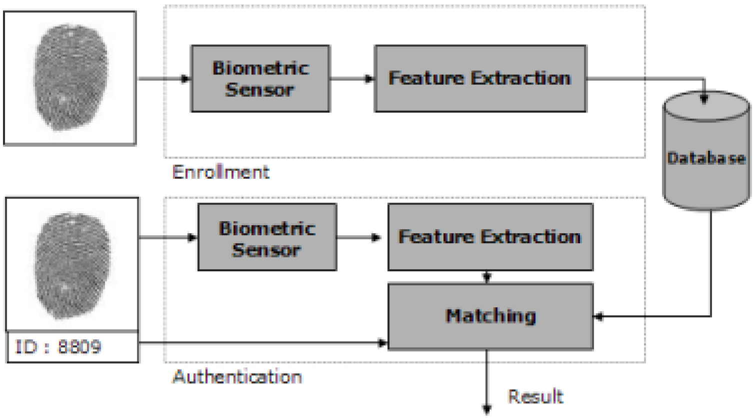


Figure 1.2: General architecture of a biometric system [3]

During enrolment, the biometrics of the user is captured and the extracted features (template) are stored in the database. During authentication, the biometrics of the user is captured again and the extracted features are compared with the ones already existing in the database to determine a match.

A typical biometric system consists of four main modules.

(i) The sensor module is used to acquire the biometric data from an individual.

(ii) The feature extraction module processes the acquired biometric data and extracts only the relevant information needed to form a new representation of the data.

(iii) The matching module compares the extracted feature set with the templates previously stored in the system database and calculates the degree of similarity (dissimilarity) between the two.

(iv) The decision module either verifies the identity claimed by the user or determines the user's identity based on the degree of similarity between the extracted features and the stored template(s). [4]

1.1.2 Verification versus Identification

When dealing with biometrics, there may be two possible types of matching to consider, verification and identification.

- Verification – A one-to-one comparison (1:1) of a biometric for a person for whom you wish to verify.
- Identification – A one-to-many comparison (1:N) of a biometric against a biometric database in attempt to identify an unknown individual.

Verification equates to "*Am I who I claim I am?"***.** When you enroll a customer for the first time, you also capture additional information such as name, phone number, or social security number. When the customer returns, they are identified through one of those pieces of information, than verified through the biometric match. Verification only proves that the person in front of you now is the one who originally enrolled.

Identification on the other hand, answers the question "*Who am I?"*. A customer is enrolled with fingerprint and additional information as noted in verification. The customer can then be identified from only their fingerprint because the system compares that fingerprint against an entire database (hence the expression one-to-many). This allows for prevention of enrolments with near-duplicate information or multiple IDs. [5] (Figure 1.3)

1.1.3 Physical and Behavioural Biometrics

- Physical Biometric- A biometric that is based on a physical trait of an individual. Examples of physical biometrics include fingerprints, hand geometry, retinal scans, and DNA. [6]

- Behavioural Biometric-A biometric that is based on a behavioural trait of an individual. Examples of behavioural biometrics include voice, signatures and keystrokes. [7]

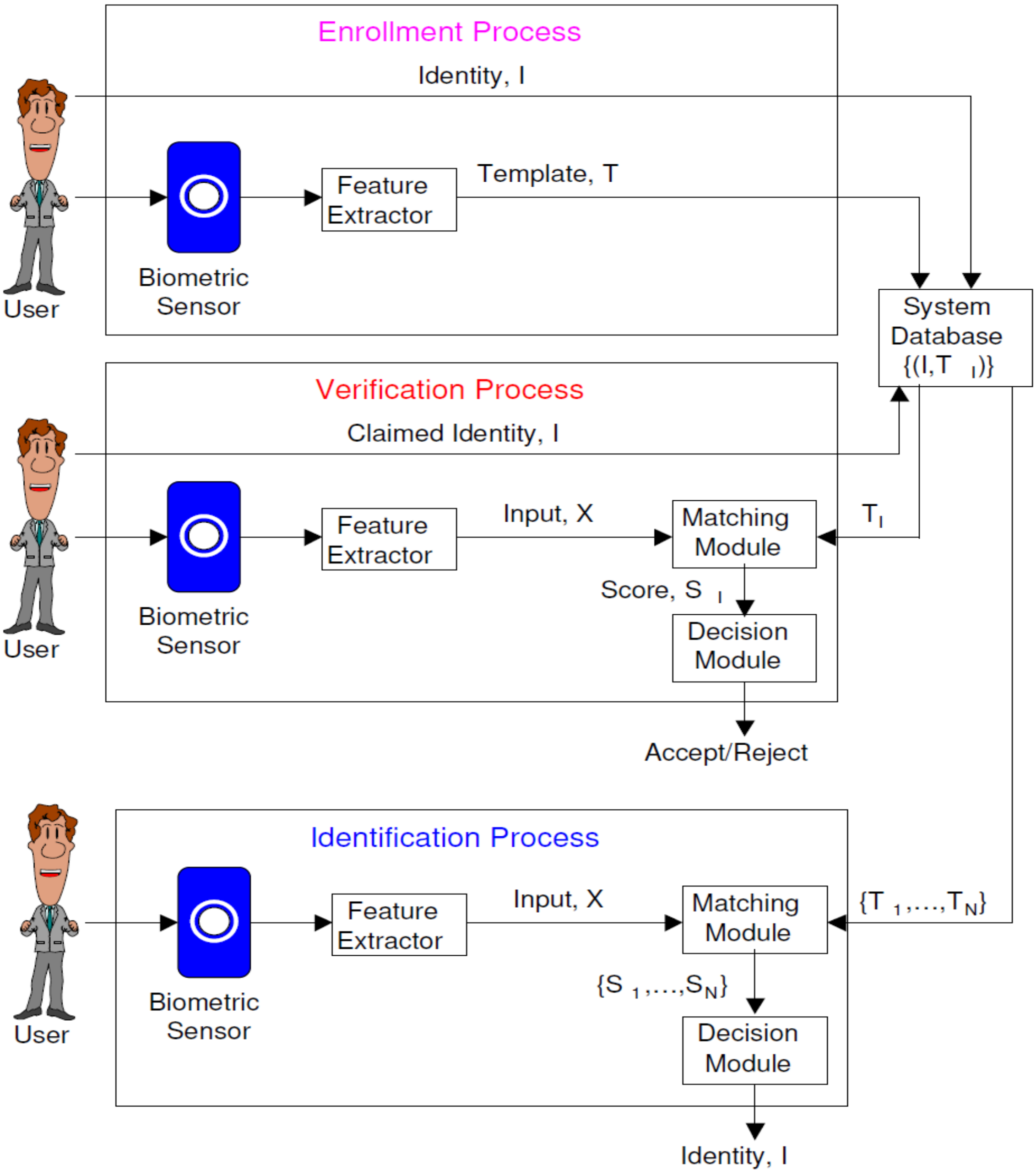


Figure 1.3: Information flow in biometric systems

In identification systems, where the input biometric sample has to be compared against many identities in the database, the use of physical characteristics such as a fingerprint or an iris leads to better performance than behavioural traits such as voice or a signature. This is because behavioural characteristics are more vulnerable to changes in the user’s emotional and physical state. Also, they may not exhibit the same level of consistency and uniqueness observed in physical traits. In addition, behavioural biometric samples (with sufficient quality and in adequate quantities) may be difficult to obtain under different operational, environmental, and geographical conditions. For example, it may be difficult to implement speaker recognition in an environment such as a factory where noisy machinery is in use.

Also, some of the major issues with voice biometrics are:

- Still imperfect technologies
- Unfamiliar to end-users
- Unproven scalability
- Voice data recording problems
- Some legal & social issues

1.1.4 Why is Voice a Good Fit for Biometric Authentication?

However, even then, voice is a good choice for recognition since a voiceprint cannot be lost or forgotten and voice biometric systems don't require specialized hardware. A speaker’s voice is extremely difficult to forge for biometrics comparison purposes, since a myriad of qualities are measured ranging from dialect and speaking style to pitch, spectral magnitudes, and format frequencies. The vibration of a user's vocal chords and the patterns created by the physical components resulting in human speech are as distinctive as fingerprints. Attempts to impersonate a voice or provide voice recordings to gain fraudulent authentication fail due to the distinctive details of the voiceprint used for comparison. While voice impersonations may sound like an exact match to the human ear, detailed mathematical analysis of the print tends to reveal vast differences. Likewise, voice recordings that sound like an exact match to the human ear most often reveal distortions caused in the recording process when measured for biometric authentication purposes. To further thwart the use of pre-recorded voiceprints, text independent directed speaker recognition systems are in place. The chances of a fraudulent user able to match a randomly generated phrase and provide a passable voice recording are remote. [8]

1.1.5 Speaker Recognition

Speaker recognition is the computing task of validating a user's claimed identity using characteristics extracted from their voices.

Speaker recognition has a history dating back some four decades and uses the acoustic features of speech that have been found to differ between individuals. These acoustic patterns reflect both anatomy (e.g., size and shape of the throat and mouth) and learned behavioural patterns (e.g., voice pitch, speaking style). The first prototype for speaker recognition was developed in 1976. Texas Instruments developed a prototype speaker recognition system that was tested by the US Air Force and MITRE Corporation. [9]

There is a difference between speaker recognition (recognizing who is speaking) and speech recognition (recognizing what is being said). These two terms are frequently confused, as is voice recognition. Voice recognition is combination of the two where it uses learned aspects of a speakers voice to determine what is being said; the system cannot recognize speech from random speakers very accurately, but it can reach high accuracy for individual voices for which it has been trained. In addition, there is a difference between the act of authentication (commonly referred to as speaker verification or speaker authentication) and identification. Speaker recognition refers to recognizing who is speaking.

1.1.6 Speaker Verification versus Speaker Identification

There are two major applications of speaker recognition technologies and methodologies. On the lines of verification and identification discussed earlier, we may say that, Speaker verification is a 1:1 match where one speaker's voice is matched to one template (also called a "voice print" or "voice model") whereas Speaker identification is a 1:N match where the voice is compared against N templates.

1.1.7 Variants of speaker recognition

Each speaker recognition system has two phases: enrolment and verification. During enrolment, the speaker's voice is recorded and typically a number of features are extracted to form a *voice print*, *template*, or *model*. In the verification phase, a speech sample or "utterance" is compared against a previously created voice print. Because of the process involved, speaker verification is usually faster than speaker identification.

Speaker recognition systems fall into two categories: text-dependent and text-independent, both of which have been discussed in detail on the next page. As is obvious, text independent speaker recognition is more complex to implement.

Text-Dependent

If the text must be the same for enrolment and verification this is called text-dependent recognition. In a text-dependent system, prompts can either be common across all speakers (e.g.: a common pass phrase) or unique. In addition, the use of shared-secrets (e.g.: passwords and PINs) or knowledge-based information can be employed in order to create a multi-factor authentication scenario.

Text-Independent

Text-independent systems are most often used for speaker identification as they require very little if any cooperation by the speaker. In this case the text during enrolment and test is different. In fact, the enrolment may happen without the user's knowledge, as in the case for many forensic applications. As text-independent technologies do not compare what was said at enrolment and verification, verification applications tend to also employ speech recognition to determine what the user is saying at the point of authentication. [10]

A block diagram of a typical speaker recognition system has been depicted on the next page in Figure 1.4.

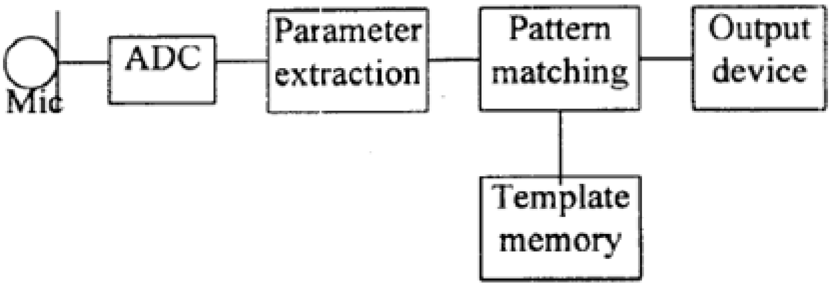


Figure 1.4: Block diagram of a typical speaker recognition system.

1.2 Prior Work:

The entire process of speaker identification may be divided into two major phases-feature extraction and classification of the extracted features. Voice features may be in different forms like MFCC[11]-[13],[19], LPC[13]-[16],[19], PLP[12],[17],[19], RASTA[15],[18]-[19] ,etc. Linear Predictive Coding (LPC) is one of the most powerful speech analysis techniques, and one of the most useful methods for encoding good quality speech at a low bit rate. It provides extremely accurate estimates of speech parameters, and is relatively efficient for computation. The Perceptual Linear Predictive (PLP) speech analysis technique [20] is based on the short term spectrum of speech. Even though the short-term spectrum of speech is subsequently modified by several psychophysìcally based spectral transformations, the PLP technique (just like most other short-term spectrum based techniques), is vulnerable when the short-term spectral values are modified by the frequency response of the communication channel. Human speech perception seems to be less sensitive to such steady state spectral factors. Relative Spectral (RASTA) methodology [12][13] makes PLP (and possibly also some other short-term spectrum based techniques) more robust to linear spectral distortions. However, out of all these, MFCC features are the most widely used for speaker recognition because of their superior performance. Many different classifiers have been used for audio recognition over the years like Nearest Neighbour, Neural Network, Gaussian Mixture Models and Hidden Markov Models [21]-[22], all of which have been discussed in detail later. Both multilayer perceptron [23]-[24] as well as radial basis function [12] neural networks have been used earlier for speaker recognition. In addition, fuzzy logic [25] and neuro-fuzzy and soft computing techniques[26] have also been applied to solve the problems of voice and speaker identification.

Ambient noise levels can impede both collection of the initial and subsequent voice samples. Noise reduction algorithms can be employed to improve accuracy, but incorrect application can have the opposite effect. Performance degradation can result from changes in behavioural attributes of the voice. Voice changes due to ageing may impact system performance over time. Capture of the biometric is seen as non-invasive.

1.3 Proposed Work

A Fuzzy Nearest Neighbour classifier has been devised for speaker recognition using MFCC features and its performance analysis is carried out with respect to multi layer perceptron Neural Network classifier, Nearest Neighbour classifier, Hidden Markov Models and Gaussian mixture clustering. The experiments have been carried out on audio samples taken from the Vid-TIMIT database. We will show that the low complexity of the proposed design allows for an implementation which works well in real-time.

1.4 Thesis Outline

Chapter 2 gives an overview of the Mel frequency cepstral coefficients that are used in this research. Chapter 3 presents a review of the different classification techniques that will be used for comparison with the system proposed in this thesis. Chapter 4 gives an overview of the proposed fuzzy classifier. Chapter 5 discusses the experimental setup. Chapter 6 outlines the results and discussions. Chapter 7 is about the conclusions and the future work.

CHAPTER 2

REVIEW OF MEL FREQUENCY CEPSTRAL COEFFICIENTS (MFCC)

2.1 Cepstral analysis, the historical father of the MFCCs.

A cepstrum is the result of taking the Inverse Fourier transform (FT) of the logarithm of the spectrum of a signal. There is a complex cepstrum, a real cepstrum, a power cepstrum, and phase cepstrum. The power cepstrum in particular finds applications in the analysis of human speech. The cepstrum is a representation used in homomorphic signal processing, to convert signals (such as a source and filter) combined by convolution into sums of their cepstra, for linear separation. This has been explained in detail below. In particular, the power cepstrum is often used as a feature vector for representing the human voice and musical signals. For these applications, the spectrum is usually first transformed using the mel scale. The result is called the mel-frequency cepstrum or MFC (its coefficients are called mel-frequency cepstral coefficients, or MFCCs). It is used for voice identification, pitch detection and much more. [27]

Cepstrum is maybe the most popular homomorphic processing because it is useful for deconvolution. To understand it, one should remember that in speech processing, the basic human speech production model adopted is a source-filter model.

Source: is related to the air expelled from the lungs. If the sound is unvoiced, like in "s" and "f", the glottis is open and the vocal cords are relaxed. If the sound is voiced, "a", "e", for example, the vocal cords vibrate and the frequency of this vibration is related to the pitch.

Filter: is responsible for giving a shape to the spectrum of the signal in order to produce different sounds. It is related to the vocal tract organs.

Roughly speaking, a good parametric representation for a speech recognition system tries to eliminate the influence of the source (the system must give the same "answer" for a high pitch female voice and for a low pitch male voice), and characterize the filter. The problem is:

source $e(n)$ and filter impulse response $h(n)$ are convoluted. Then we need deconvolution in speech recognition applications.

Mathematically: In the time domain, convolution: source * filter = speech,

$$e(n) * h(n) = x(n). \qquad (2.1)$$

In the frequency domain, multiplication: source x filter = speech,

$$E(z)\,H(z) = X(z). \qquad (2.2)$$

How can we make the deconvolution ? *Cepstral analysis* is an alternative.

Working in the frequency domain, use the logarithm to transform the multiplication in (2.2) into a summation (obs: log ab = log a + log b). It is not easy to separate (to filter) things that are multiplied as in (2.2), but it is easy to design filters to separate things that are parcels of a sum as below:

C(z) = log X(z) = log E(z) + log H(z). (2.3)

We hope that H(z) is mainly composed by low frequencies and E(z) has most of its energy in higher frequencies, in a way that a simple low-pass filter can separate H(z) from E(z) if we were dealing with E(z) + H(z). In fact, let us suppose for the sake of simplicity that we have, instead of (2.3), the following equation:

Co(z) = E(z) + H(z). (2.4)

We could use a linear filter to eliminate E(z) and then calculate the Z-inverse transform to get a time-sequence co(z). Notice that in this case, co(z) would have dimension of time (seconds, for example). Having said that, let us now face our problem: the log operation in (2.3). Log is a nonlinear operation and it can "create" new frequencies. For example, expanding the log of a cosine in Taylor series shows that harmonics are created. So, even if E(z) and H(z) are well separated in the frequency domain, log E(z) and log H(z) could eventually have considerable overlap. Fortunately, that is not the case in practice for speech processing. The other point is that, because of the log operation, the Z-inverse of C(z) in (2.3) has NOT the dimension of time as in (2.4). We call *cepstrum* the Z-inverse of C(z) and its dimension is *quefrency* (a time domain parameter).

2.1.1 Liftering

A filter that operates on a cepstrum might be called a lifter. Liftering is applied according to the following equation, where c_n is the nth feature element in the feature vector.

$$c_n' = \left(1 + \tfrac{N}{2} sin \tfrac{\pi n}{N}\right) c_n \qquad (2.5)$$

2.1.2 Pre-Emphasis

It is common practice that before extraction of MFCC features, pre-emphasis is carried out to reduce the high frequency falloff. In processing electronic audio signals, pre-emphasis refers to a system process designed to increase (within a frequency band) the magnitude of some (usually higher) frequencies with respect to the magnitude of other (usually lower) frequencies in order to improve the overall signal-to-noise ratio by minimizing the adverse effects of such phenomena as attenuation distortion or saturation of recording media in subsequent parts of the system.

The first order difference equation:

$$s_n^{'} = s_n - \alpha s_{n-1} \qquad (2.6)$$

is applied on a window of input samples. Here α is the pre-emphasis filter coefficient in the range *[0, 1)*. [28]

2.1.3 Hamming Window

Then, Hamming window is applied to minimize spectral leakage. In MATLAB, w = hamming(L) returns an L-point symmetric Hamming window in the column vector w. L should be a positive integer. The coefficients of a Hamming window are computed from the following equation.

$$w(n) = 0.54 - 0.46\cos\left(2\pi\,\frac{n}{N}\right),\ 0 \le n \le N \qquad (2.7)$$

The window length is $L = N + 1$. [29]

2.2 Introduction to Mel Frequency Cepstral Coefficients (MFCC)

2.2.1 History

Paul Mermelstein is typically credited with the development of the Mel Frequency Cepstral (MFC). Mermelstein credits Bridle and Brown for the idea: Bridle and Brown used a set of 19 weighted spectrum-shape coefficients given by the cosine transform of the outputs of a set of non-uniformly spaced bandpass filters. The filter spacing is chosen to be logarithmic above

1 kHz and the filter bandwidths are increased there as well. We will, therefore, call these the mel-based cepstral parameters.

2.2.2 Basics

In sound processing, the mel-frequency cepstrum (MFC) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency.

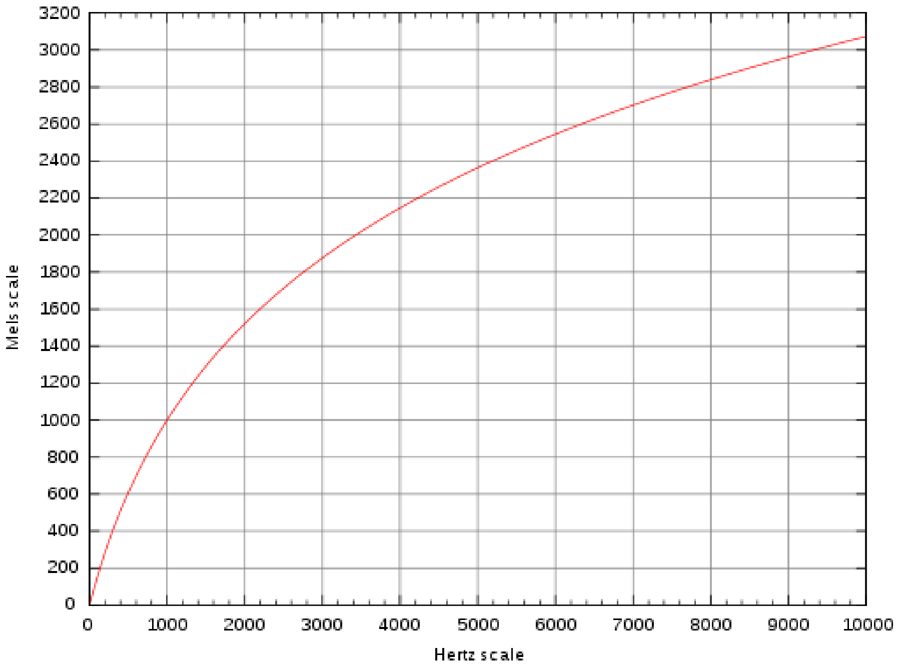


Figure 2.1: Plot of pitches mels versus vs Hertz

The mel scale, named by Stevens, Volkman and Newman in 1937 is a perceptual scale of pitches judged by listeners to be equal in distance from one another. The reference point between this scale and normal frequency measurement is defined by assigning a perceptual pitch of 1000 mels to a 1000 Hz tone, 40 dB above the listener's threshold. Above about 500 Hz, larger and larger intervals are judged by listeners to produce equal pitch increments. As a result, four octaves on the hertz scale above 500 Hz are judged to comprise about two octaves on the mel scale. The name mel comes from the word melody to indicate that the scale is based on pitch comparisons. A plot of pitch in mels and hertz has been shown in Fig. 2.1 given above.

A popular formula to convert f hertz into m mel is: [30]

$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \qquad (2.8)$$

Mel-frequency cepstral coefficients (MFCCs) are coefficients that collectively make up an MFC. They are derived from a type of cepstral representation of the audio clip (a nonlinear "spectrum-of-a-spectrum"). The difference between the cepstrum and the mel-frequency cepstrum is that in the MFC, the frequency bands are equally spaced on the mel scale, which approximates the human auditory system's response more closely than the linearly-spaced frequency bands used in the normal cepstrum. This frequency warping can allow for better representation of sound, for example, in audio compression.

MFCCs are commonly derived as follows:

1. Take the Fourier transform of (a windowed excerpt of) a signal.
2. Map the powers of the spectrum obtained above onto the mel scale, using triangular overlapping windows.
3. Take the logs of the powers at each of the mel frequencies.
4. Take the discrete cosine transform of the list of mel log powers, as if it were a signal.
5. The MFCCs are the amplitudes of the resulting spectrum.

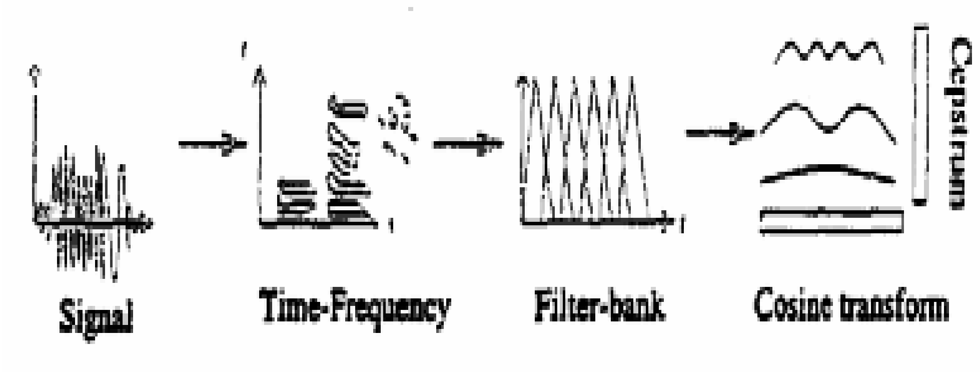


Figure 2.2: Pictorial representation of mel-frequency cepstrum (MFCC) calculation[28]

There can be variations on this process, for example, differences in the shape or spacing of the windows used to map the scale. The European Telecommunications Standards Institute in the early 2000s defined a standardised MFCC algorithm to be used in mobile phones. [31]

2.2.3 Delta and Acceleration Coefficients

The performance of a speech recognition system can be greatly enhanced by adding time derivatives to the basic static parameters. The first order regression coefficients (referred to as delta coefficients) are appended, and the second order regression coefficients (referred to as acceleration coefficients) are appended. The first order regression coefficients (delta coefficients) are computed by the following regression equation:

$$d_i = \frac{\sum_{n=1}^{N} n(c_{n+i} - c_{n-i})}{2 \sum_{n=1}^{N} n^2} \qquad (2.9)$$

where d_i is the delta coefficient at frame *i* computed in terms of the corresponding basic coeffecients c_{n+i} to c_{n-i} . The same equation is used to compute the acceleration coefficients by replacing the basic coefficients with the delta coefficients.

2.2.4 Block Diagram

To summarize, we may state that, the block diagrams for calculating MFCCs is given below.

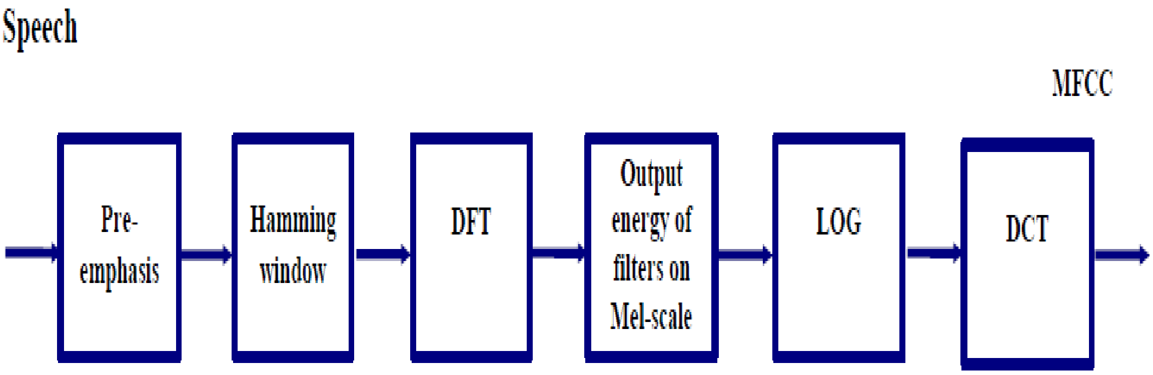


Figure 2.3 : Block Diagram for MFCC calculation

The MFCC Features along with their corresponding delta and acceleration form a 39 dimensional audio feature vector for each frame in a given sample. We use the first 13 MFCC Features (12+ Frame Energy) and their velocity(delta) and accelerations(delta-delta) to form a 39-dimensional feature vector. These are used as voice features for many speech/speaker recognition tasks. The proposed system will also make use of these 39 dimensional audio feature vector to represent each frame in a given audio sample[28].

2.2.5 Applications of MFCC

MFCCs are commonly used as features in speech recognition systems, such as the systems which can automatically recognize numbers spoken into a telephone. They are also common in speaker recognition, which is the task of recognizing people from their voices. MFCCs are also increasingly finding uses in music information retrieval applications such as genre classification, audio similarity measures, etc. [31]

CHAPTER 3

REVIEW OF EXISTING CLASSIFIERS

In this chapter we will discuss some of the already existing techniques that are in place for speaker recognition and which have been used in this research for comparison with the results obtained by the proposed system.

3.1 Neural Network Classifier

3.1.1 History

The earliest work in neural computing goes back to the 1940's when McCulloch and Pitts introduced the first neural network computing model. In the 1950's, Rosenblatt's work resulted in a two-layer network, the perceptron, which was capable of learning certain classifications by adjusting connection weights. Although the perceptron was successful in classifying certain patterns, it had a number of limitations. The perceptron was not able to solve the classic XOR (exclusive or) problem. Such limitations led to the decline of the field of neural networks. However, the perceptron had laid foundations for later work in neural computing. In the early 1980's, researchers showed renewed interest in neural networks. Recent work includes Boltzmann machines, Hopfield nets, competitive learning models, multilayer networks, and adaptive resonance theory models. [32]

3.1.2 Basics

Neural networks are composed of simple elements operating in parallel. These elements are inspired by biological nervous systems. As in nature, the connections between elements largely determine the network function. A neural network may be trained to perform a particular function by adjusting the values of the connections (weights) between elements. Typically, neural networks are adjusted, or trained, so that a particular input leads to a specific target output. The figure given on the next page illustrates such a situation. There, the network is adjusted, based on a comparison of the output and the target, until the network output matches the target. Typically, many such input/target pairs are needed to train a network.

Neural networks have been trained to perform complex functions in various fields, including pattern recognition, identification, classification, speech, vision, and control systems. Neural networks can also be trained to solve problems that are difficult for conventional computers or human beings. MATLAB provides four graphical tools for training neural networks to solve problems in function fitting, pattern recognition, clustering, and time series. In the

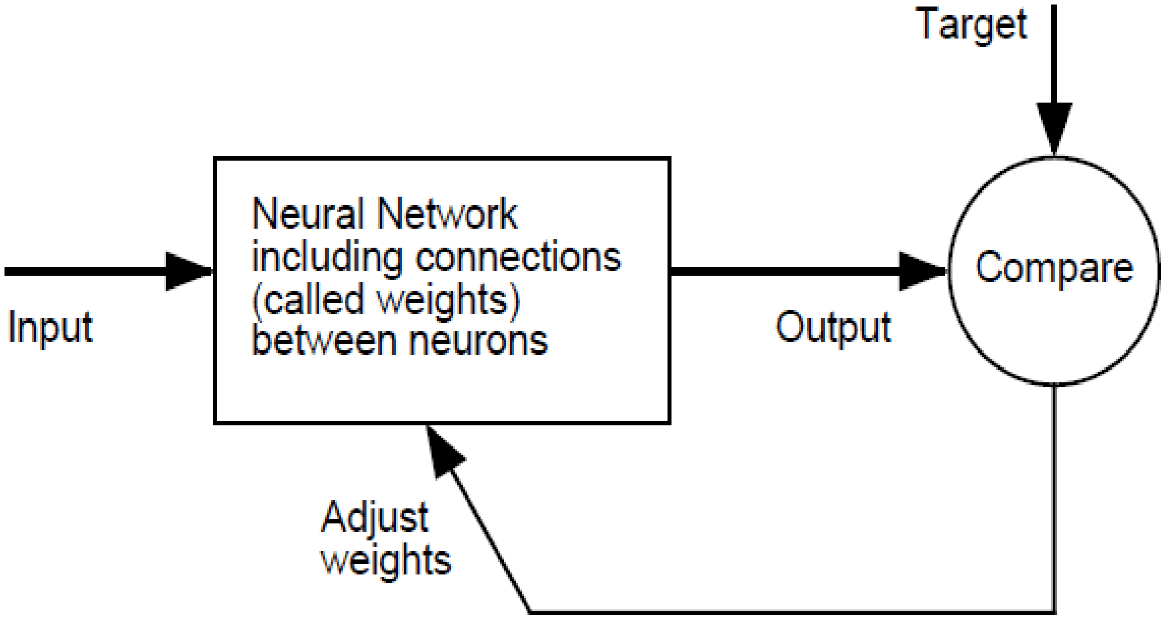

Target
Neural Network including connections (called weights) between neurons
Input
Output
Compare
Adjust weights

Figure 3.1 : Working of a Neural Network

remaining sections of this chapter, you will follow the standard steps for designing neural networks to solve problems in four application areas: function fitting, pattern recognition, clustering, and time series analysis. The work flow for any of these problems has six primary steps. (Data collection, while important, generally occurs outside the MATLAB environment, so it is step 0.)

0 Collect data.

1 Create the network.

2 Configure the network.

3 Initialize the weights and biases.

4 Train the network.

5 Validate the network.

6 Use the network.

3.1.3 Recognizing Patterns

In addition to function fitting, neural networks are also good at recognizing patterns. For example, suppose you want to classify a tumor as benign or malignant, based on uniformity of cell size, clump thickness, mitosis, etc. You have 699 example cases for which you have 9 items of data and the correct classification as benign or malignant.

The nprtool GUI in MATLAB as described in Using the Neural Network Pattern Recognition Tool may be used to recognize patterns. While training, the input vectors and target vectors will be randomly divided into three sets as follows:

•70% are used for training.

•15% are used to validate that the network is generalizing and to stop training before overfitting.

•The last 15% are used as a completely independent test of network generalization.

3.1.4 Defining a Problem

To define a pattern recognition problem, arrange a set of Q input vectors as columns in a matrix. Then arrange another set of Q target vectors so that they indicate the classes to which the input vectors are assigned. There are two approaches to creating the target vectors. One approach can be used when there are only two classes; you set each scalar target value to either 1 or 0, indicating which class the corresponding input belongs to. Alternately, target vectors can have *N* elements, where for each target vector, one element is 1 and the others are 0. This defines a problem where inputs are to be classified into *N* different classes. The results show very good recognition. If even more accurate results are needed, any of the following approaches may be used:

- Increase the number of hidden neurons.
- Increase the number of training vectors. [33]

3.2 K-NN Classifier

3.2.1 History

K-nearest-neighbour (k-NN) classification is one of the most fundamental and simple classification methods and should be one of the first choices for a classification study when there is little or no prior knowledge about the distribution of the data. K-nearest-neighbour classification was developed from the need to perform discriminant analysis when reliable parametric estimates of probability densities are unknown or difficult to determine. In an unpublished US Air Force School of Aviation Medicine report in 1951, Fix and Hodges introduced a non-parametric method for pattern classification that has since become known the k-nearest neighbour rule (Fix & Hodges, 1951). Later in 1967, some of the formal properties of the k-nearest-neighbour rule were worked out; for instance it was shown that for k=1 and $n\rightarrow\infty$ the k-nearest-neighbour classification error is bounded above by twice the Bayes error rate (Cover & Hart, 1967). Once such formal properties of k-nearest-neighbour classification were established, a long line of investigation ensued including new rejection approaches (Hellman, 1970), refinements with respect to Bayes error rate (Fukunaga & Hostetler, 1975), distance weighted approaches (Dudani, 1976; Bailey & Jain, 1978),soft computing (Bermejo & Cabestany, 2000) methods and fuzzy methods (Jozwik, 1983; Keller et al, 1985). [34]

3.2.2 Basics

In pattern recognition, the *k*-nearest neighbour algorithm (*k*-NN) is a method for classifying objects based on closest training examples in the feature space. *k*-NN is a type of instance-based learning, or lazy learning where the function is only approximated locally and all computation is deferred until classification. The *k*-nearest neighbour algorithm is amongst the simplest of all machine learning algorithms: an object is classified by a majority vote of its neighbours, with the object being assigned to the class most common amongst its *k* nearest neighbours (*k* is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of its nearest neighbour. The neighbours are taken from a set of objects for which the correct classification is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. An example of k-NN classification has been illustrated in the figure 3.2 given below. The test sample i.e., the green circle has to be classified to one of the classes- the red triangles or the blue squares. Now, if k=3, it means classification has to be done looking at 3 nearest neighbours. So, the test sample is assigned to the class red triangles, because there are two triangles and only one square inside the inner circle. On the other hand, if k=5, it means that classification has to carried out taking into consideration the 5 nearest neighbours. The test sample is classified to the class of blue squares, as there are 3 squares and only 2 triangles inside the outer circle.

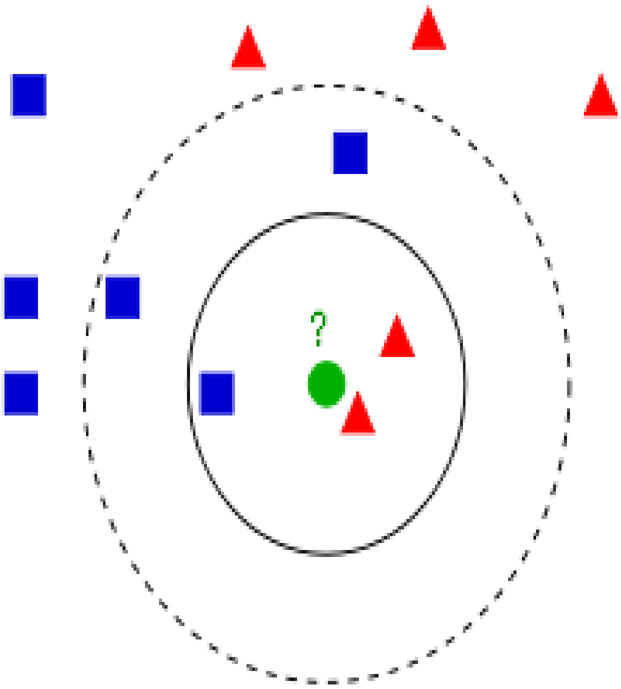


Figure 3.2 : Classification using the k-Nearest Neighbour Technique

---

The training examples are vectors in a multidimensional feature space, each with a class label. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples. In the classification phase, *k* is a user-defined constant, and an unlabeled vector (a query or test point) is classified by assigning the label which is most frequent among the *k* training samples nearest to that query point. Usually Euclidean distance is used as the distance metric. In cases such as text classification, another metric such as the Hamming distance may be used. [35]

For any classification to be carried out using the k-Nearest Neighbour classifier the distance metric has to be specified explicitly.

Choices are:

- 'euclidean' — Euclidean distance
- 'cityblock' — Sum of absolute differences
- 'cosine' — One minus the cosine of the included angle between points (treated as vectors)
- 'correlation' — One minus the sample correlation between points (treated as sequences of values)
- 'hamming' — Percentage of bits that differ (suitable only for binary data)

Also, the rule that is used to decide how the sample has to be classified needs to be specified. It may be one of the following:

'nearest' — Majority rule with nearest point tie-break
'random' — Majority rule with random point tie-break
'consensus' — Consensus rule [36]

3.2.3 Parameter Selection

The best choice of *k* depends upon the data; generally, larger values of *k* reduce the effect of noise on the classification, but make boundaries between classes less distinct The special case where the class is predicted to be the class of the closest training sample (i.e. when *k* = 1) is called the nearest neighbour algorithm.

The accuracy of the *k*-NN algorithm can be severely degraded by the presence of noisy or irrelevant features, or if the feature scales are not consistent with their importance.

3.3 Hidden Markov Models

Hidden Markov Models (HMM) are stochastic methods to model temporal and sequence data. They are especially known for their application in temporal pattern recognition such as speech, handwriting, gesture recognition, musical score following, partial discharges and bioinformatics.

3.3.1 History

Hidden Markov Models were first described in a series of statistical papers by Leonard E. Baum and other authors in the second half of the 1960s. One of the first applications of HMMs was speech recognition, starting in the mid-1970s. Indeed, one of the most comprehensive explanations on the topic was published in “A Tutorial On Hidden Markov Models And Selected Applications in Speech Recognition”, by Lawrence R. Rabiner in 1989. In the second half of the 1980s, HMMs began to be applied to the analysis of biological sequences, in particular DNA. Since then, they have become ubiquitous in the field of bioinformatics.

3.3.2 Basics

Dynamical systems of discrete nature assumed to be governed by a Markov chain emits a sequence of observable outputs. Under the Markov assumption, it is also assumed that the latest output depends only on the current state of the system. Such states are often not known from the observer when only the output values are observable.

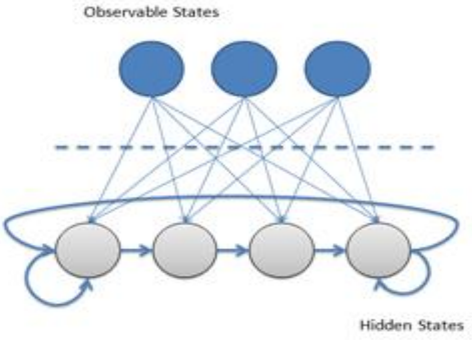


Figure 3.3: Output and Hidden States in an HMM

Hidden Markov Models attempt to model such systems and allow, among other things, (1) to infer the most likely sequence of states that produced a given output sequence, to (2) infer which will be the most likely next state (and thus predicting the next output) and (3) calculate the probability that a given sequence of outputs originated from the system (allowing the use of hidden Markov models for sequence classification). The “hidden” in Hidden Markov Models comes from the fact that the observer does not know in which state the system may be in, but has only a probabilistic insight on where it should be.

3.3.3 Notation

Traditionally, HMMs have been defined by the following quintuple:

$$\lambda = (N, M, A, B, \pi)$$

where

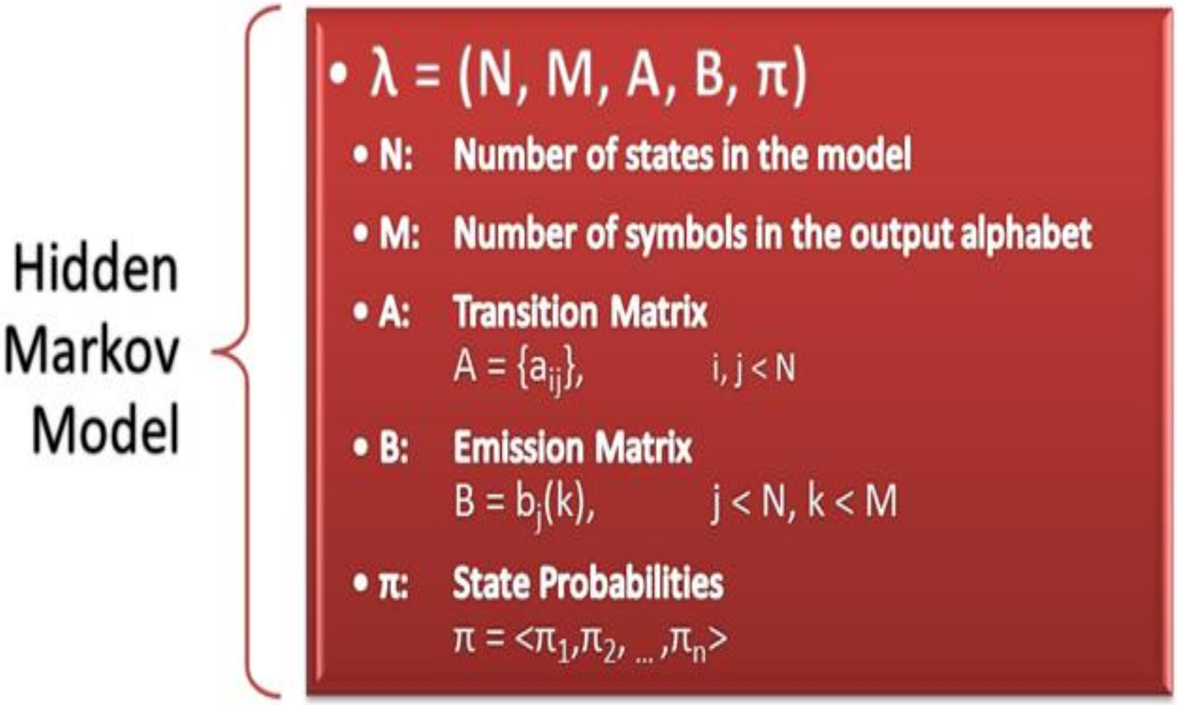
N is the number of states for the model
M is the number of distinct observations symbols per state, i.e. the discrete alphabet size.

A is the NxN state transition probability distribution given in the form of a matrix A = $\{a_{ij}\}$
B is the NxM observation symbol probability distribution given in the form of a matrix B = $\{b_j(k)\}$
π is the initial state distribution vector $\pi = \{\pi_i\}$

Note that, if we opt out the structure parameters M and N we have the more often used compact notation

$$\lambda = (A, B, \pi)$$

Hidden Markov Models can be seen as finite state machines where for each sequence unit observation there is a state transition and, for each state, there is a output symbol emission. The picture below summarizes the overall definition of a HMM.



Hidden Markov Model

- λ = (N, M, A, B, π)
  - N: Number of states in the model
  - M: Number of symbols in the output alphabet
  - A: Transition Matrix
    $A = \{a_{ij}\}$, $i, j < N$
  - B: Emission Matrix
    $B = b_j(k)$, $j < N, k < M$
  - π: State Probabilities
    $\pi = <\pi_1, \pi_2, \ldots, \pi_n>$

Figure 3.4 : Overall definition of HMM

3.3.4 Canonical problems

There are three canonical problems associated with hidden Markov models, given the parameters of the model, compute the probability of a particular output sequence. This requires summation over all possible state sequences, but can be done efficiently using the Forward algorithm, which is a form of dynamic programming. Given the parameters of the model and a particular output sequence, find the state sequence that is most likely to have generated that output sequence. This requires finding a maximum over all possible state sequences, but can similarly be solved efficiently by the Viterbi algorithm. Given an output sequence or a set of such sequences, find the most likely set of state transition and output probabilities. In other words, derive the maximum likelihood estimate of the parameters of the HMM given a dataset of output sequences. No tractable algorithm is known for solving this problem exactly, but a local maximum likelihood can be derived efficiently using the Baum-Welch algorithm or the Baldi-Chauvin algorithm. The Baum-Welch algorithm is an example of a forward-backward algorithm, and is a special case of the Expectation-maximization algorithm. The solution for those problems are exactly what makes Hidden Markov Models useful. The ability to learn from the data and then become able to make predictions and able to classify sequences is nothing but applied machine learning.

3.3.5 Choosing the structure

Choosing the structure for a hidden Markov model is not always obvious. The number of states depend on the application and to what interpretation one is willing to give to the hidden states. Some domain knowledge is required to build a suitable model and also to choose the initial parameters that an HMM can take. There is also some trial and error involved, and there are sometimes complex tradeoffs that have to be made between model complexity and difficulty of learning, just as is the case with most machine learning techniques.

3.4 Gaussian Mixture Clustering

Clustering is used to process M distinct data sets in a single pass. Mixture models for each data set are extracted and stored (total M mixture models) in a signal parameter file. This is useful for applications such as segmentation when each mixture model represents one of M distinct classes that must be modelled. In order to run the algorithm a data file must be created for each of the M data sets. Each data file contains a series of vectors in ASCII floating point format and on separate lines. Each vector should be a sample from the multivariate distribution of interest. Then, these data vectors will be used to estimate a Gaussian mixture model that best fits the sample data in the corresponding file. The Gaussian mixture model is formed by adding together multivariate Gaussian distributions each with different mean and covariance. Each of these component distributions is a cluster (or subclass) of the distribution. After a Gaussian mixture model has been extracted for each data set, a file will be generated which contains all the parameters of all M Gaussian mixture distributions. The basic operations performed have been explained. The algorithm is started by initializing with a set of cluster parameters and a user selected number of clusters. The cluster means are generated by selecting the appropriate number of samples from the training data, and the cluster covariances are set to all be equal to the covariance of the complete data set. After this initialization, the algorithm enters a loop in which clusters are combined (or eliminated when empty) until only one cluster remains. [37]

3.4.1 Introduction to Gaussian Mixture Models

Gaussian mixture models are formed by combining multivariate normal density components. Data is fitted using an expectation maximization (EM) algorithm, which assigns posterior probabilities to each component density with respect to each observation. Gaussian mixture models are often used for data clustering. Clusters are assigned by selecting the component that maximizes the posterior probability. Like *k*-means clustering, Gaussian mixture modelling uses an iterative algorithm that converges to a local optimum. Clustering using Gaussian mixture models is sometimes considered a soft clustering method. The posterior probabilities for each point indicate that each data point has some probability of belonging to each cluster. Here we deal with the use of Gaussian mixture models in clustering. [38]

3.4.2 Clustering with Gaussian Mixtures

Gaussian mixture distributions can be used for clustering data, by realizing that the multivariate normal components of the fitted model can represent clusters. To demonstrate the process,

1) First generate some simulated data from a mixture of two bivariate Gaussian distributions.

2) Fit a two-component Gaussian mixture distribution. Here, we know the correct number of components to use. In practice, with real data, this decision would require comparing models with different numbers of components.

3) Plot the estimated probability density contours for the two-component mixture distribution. Let us suppose that the two bivariate normal components overlap, but their peaks are distinct as shown in Fig 3.5. This suggests that the data could reasonably be divided into two clusters.

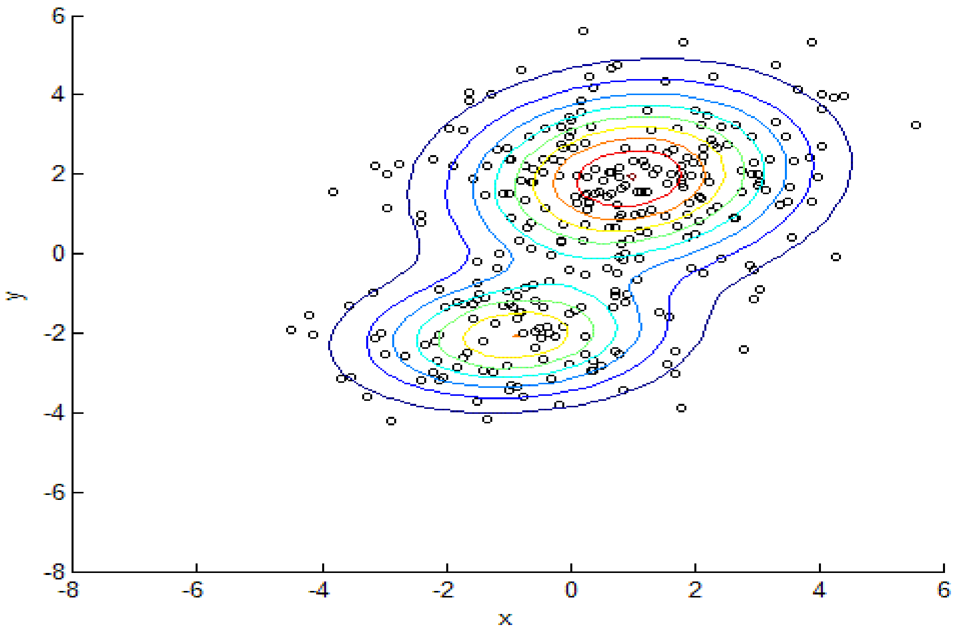


Figure 3.5: Probability density contours for two component mixture distribution

4) Partition the data into clusters. This will assign each point to one of the two components in the mixture distribution.

Each cluster corresponds to one of the bivariate normal components in the mixture distribution. Points are assigned to clusters based on the estimated posterior probability that a point came from a component; each point is assigned to the cluster corresponding to the highest posterior probability [37].

CHAPTER 4

PROPOSED FUZZY CLASSIFIER

4.1 Introduction

Speaker Classification is the most important step in speaker identification. A speaker classifier compares two sets of features originating from two different audio samples and determines whether or not they represent the same speaker. Speaker classification is an extremely difficult problem mainly due to large intra-class variations that may exist in the audio samples taken from the same speaker. The intra-class variations are mainly due to the fact that voice being a behavioural biometric, a person’s voice/audio sample captured at any point in time may be affected by a number of factors such as the person’s mood, person’s health condition (like a person’s voice when he/she is suffering from cold may be slightly different from his normal voice), presence of noise in the background, etc. A fuzzy classifier is built for text independent speaker recognition using MFCC features. The fuzzy classifier is based on the gaussian membership function and its performance analysis is done with respect to some of the other classifiers like Neural Network classifier, Nearest Neighbour classifier, Hidden Markov Model (HMM) classifier and Gaussian Mixture Clustering. In the past, similar work has been carried out for the recognition of handwritten hindi numerals[39].

4.2 Fuzzy Logic

Fuzzy logic is a form of many-valued logic or probabilistic logic; it deals with reasoning that is approximate rather than fixed and exact. In contrast with traditional logic theory, where binary sets have two-valued logic: true or false, fuzzy logic variables may have a truth value that ranges in degree between 0 and 1. Fuzzy logic has been extended to handle the concept of partial truth, where the truth value may range between completely true and completely false. For example, the statement, *today is sunny*, might be 100% true if there are no clouds, 80% true if there are a few clouds, 50% true if it's hazy and 0% true if it rains all day. Fuzzy logic began with the 1965 proposal of fuzzy set theory by Lotfi Zadeh. Fuzzy logic has been applied to many fields, from control theory to artificial intelligence. Tremendous success achieved in numerous fields by the use of fuzzy logic laid the motivation for the use of fuzzy logic for speaker identification.

4.2.1 Membership Functions and the Gaussian Membership Function

The membership function of a fuzzy set is a generalization of the indicator function in classical sets. In fuzzy logic, it represents the degree of truth as an extension of valuation. For any set X, a membership function on X is any function from X to the real unit interval [0,1]. The membership function which represents a fuzzy set $\tilde{A}$ is usually denoted by μ_A. For an element x of X, the value $\mu_A(x)$ is called the membership degree of x in the fuzzy set $\tilde{A}$. The membership degree $\mu_A(x)$ quantifies the grade of membership of the element x to the fuzzy set $\tilde{A}$. The value 0 means that x is not a member of the fuzzy set; the value 1 means that x is fully a member of the fuzzy set. The values between 0 and 1 characterize fuzzy members, which belong to the fuzzy set only partially.

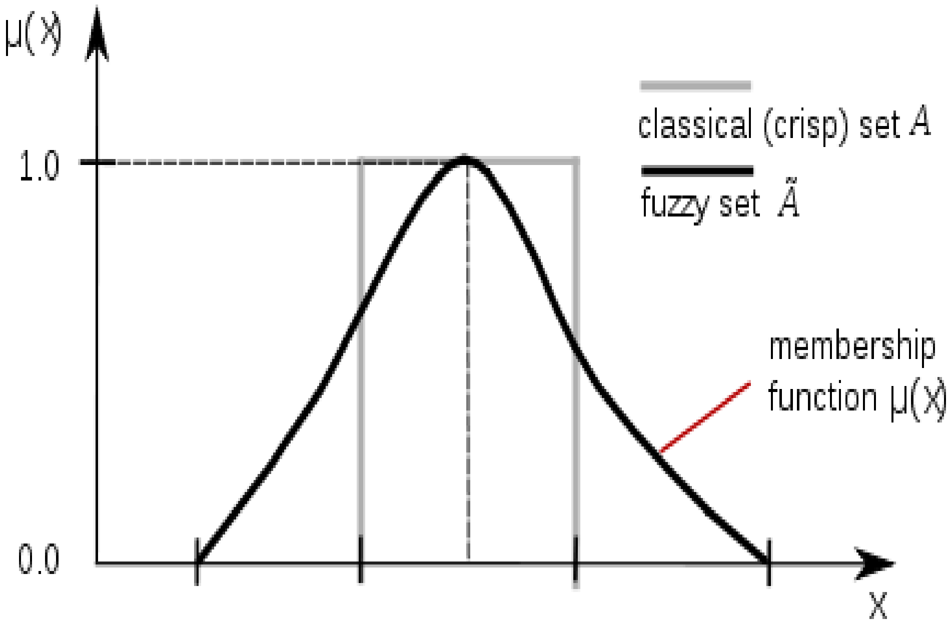


Figure 4.1: Membership function of a fuzzy set

The Gaussian membership function will be used in the proposed scheme. An overview of the Gaussian membership function is given on the next page. A plot of the Gaussian membership function is presented in Fig 4.2.

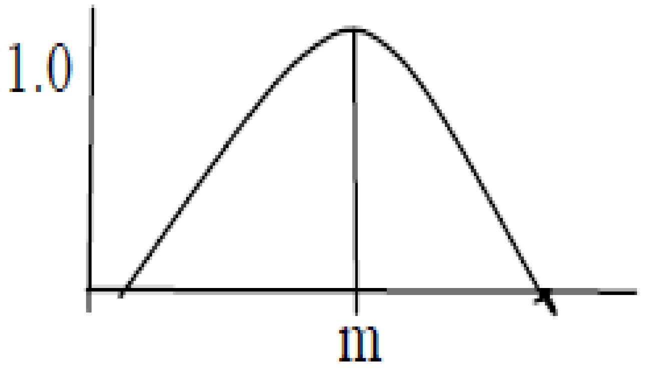


Figure 4.2 : The Gaussian membership function

The Gaussian curve is given by

$$f(x) = \exp\left(\frac{-0.5(x-c)^2}{\sigma^2}\right) \tag{4.1}$$

where *c* is the mean and σ is the variance and control the centre and width of the membership function respectively.

Gaussian membership function has been selected because of the following advantages:

(i) The Gaussian functions facilitate obtaining smooth continuously differentiable hypersurfaces of a fuzzy model.

(ii) The Gaussian functions facilitate theoretical analysis of fuzzy systems as they are continuously differentiable and infinitely differentiable. [40]

4.3 Fuzzy Classification

Fuzzy classification is an application of fuzzy theory. Fuzzy classification is the process of grouping elements into a fuzzy set which allows its members to have different grades of membership (membership function) in the interval [0, 1]. One possible definition of a fuzzy classifier is given in (Kuncheva 2000) as 'any classifier that uses fuzzy sets or fuzzy logic in the course of its training or operation.' In fuzzy classification an instance can belong to different classes with different membership degrees; conventionally the sum of the membership values of each single instance must be unitary. The main advantage of fuzzy classification based method includes its applicability for very complex processes.

4.3.1 Why fuzzy classifiers?

A classifier is an algorithm that assigns a class label to an object, based on the object description. It is also said that the classifier predicts the class label. The object description comes in the form of a vector containing values of the features (attributes) deemed to be relevant for the classification task. Typically, the classifier learns to predict class labels using a training algorithm and a training data set. When a training data set is not available, a classifier can be designed from prior knowledge and expertise. Once trained, the classifier is ready for operation on unseen objects.

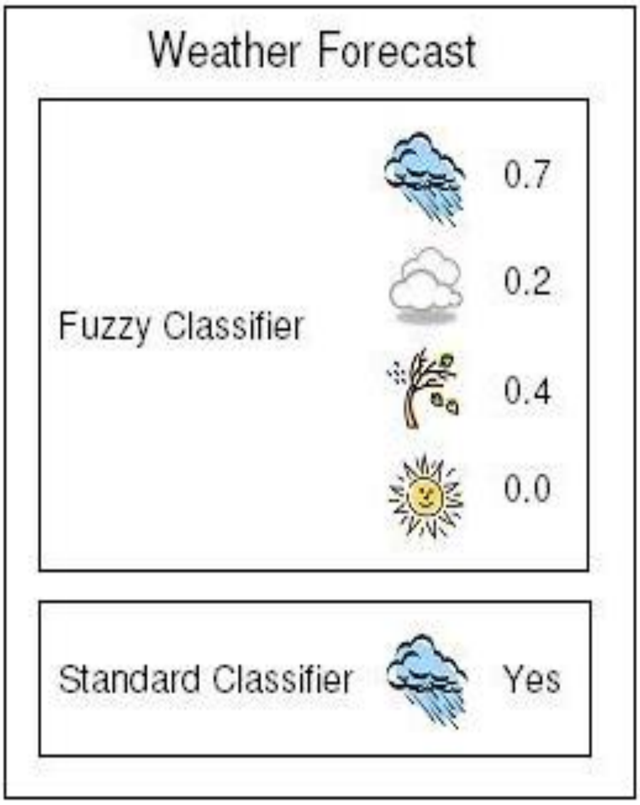


Figure 4.3**:** Fuzzy classifier produces soft class labels.

Classification belongs to the general area of pattern recognition and machine learning.

- Soft labelling**:** The standard assumption in pattern recognition is that the classes are mutually exclusive. However, this may not always be the case. A standard classifier will assign a single crisp label (rain). A fuzzy classifier can assign degrees of membership (soft labels) in all four classes {rain, clouds, wind, sunshine}, accounting for the possibility of winds and cloudy weather throughout the day. This has been represented in Fig 4.3. A standard classifier can output posterior probabilities, and offer soft labelling too. However, a probability of, say, 0.2 for cloudy weather means that there is 20% chance that tomorrow will be cloudy. A probabilistic model would also assume that the four classes form a full group, i.e., snow, blizzards or thunderstorms must be subsumed by one of the existing four classes. Soft labelling is free from this assumption. A fuzzy classifier (D), producing soft labels can be perceived as a function approximator $D:F\rightarrow[0,1]^c$, where F is the feature space where the object descriptions live, and c is the number of classes. While tuning such a function approximator outside the classification scenario would be very difficult, fuzzy classifiers may provide a solution that is both intuitive and useful.

- Interpretability: Automatic classification in most challenging applications such as medical diagnosis has been sidelined due to ethical, political or legal reasons, and mostly due to the black box philosophy underpinning classical pattern recognition. Fuzzy classifiers are often designed to be transparent, i.e., steps and logic statements leading to the class prediction are traceable and comprehensible.

- Limited data, available expertise**:** Examples include predicting and classification of rare diseases, oil depositions, terrorist activities, natural disasters. Fuzzy classifiers can be built using expert opinion, data or both.

4.4 Design of the Proposed Fuzzy Classifier

The proposed fuzzy classifier, based on the MFCC features extracted from an audio sample, will classify any presented audio sample into one of the classes using the mean dataset and the test dataset and based on the fuzzy classification algorithm developed.

The procedure used by the algorithm is first explained in a nutshell. For any sample to be classified, degree of membership of each frame in the sample to each of the classes will be calculated. The degree of membership will be calculated using the Gaussian membership function. The basic procedure to calculate degree of membership of a frame in the test sample to a particular class is as follows: Degree of membership of the test frame to each frame in the mean dataset (which represents a sample) is calculated. Then, the average of all the membership degrees corresponding to the samples of a particular class is the membership degree of the frame to the class. Now, this procedure is repeated and degree of membership of each frame in the test dataset to all the classes is calculated. As, explained earlier, degree of membership will be a value between 0 and 1, that will indicate the membership of the frame to a particular class. Then, the frame will be considered to belong to the class to which its degree of membership is the highest, and will be assigned to that particular class. An audio sample will be assigned to the class to which maximum of its frames belong. The step-by-step details have been given in the sections following.

4.4.1 Fuzziness in the classifier

The proposed classifier is a "fuzzy prototype-based" classifier. It is a 'fuzzy prototype based' classifier because a 'frame prototype' for each training sample is obtained. The prototype of each sample contains a single frame. This prototype is the mean calculated from all the test frames in the sample. Then the fuzzy prototype corresponding to a sample is used for calculation of membership degree of a given test sample to the particular training sample. The proposed classifier is a type of fuzzy nearest-neighbour classifier. Usually prototype based speaker identification systems use a 'speaker prototype'. But, here we have worked with 'frame prototype' obtained from the frames of a sample instead of speaker prototype because the different training samples have been obtained under different environments/conditions, over a few days, and we want to maintain that distinction between the training samples.

4.4.2 Setting up the training and test databases

As explained earlier, experiments have been conducted on a total of 43 speakers and out of the ten audio samples per speaker, M are used for training. Each audio sample consists of a fixed number of frames. MFCC features have to be extracted for each frame in each audio sample. Assuming each audio sample to be composed of 50 frames, the total size of the training dataset= (43*M*50)*39. It was explained in chapter 2 how we achieved 39 dimensional feature vector. We will be using the term ‘original training dataset’ to refer to this dataset in the following sections of this work.

4.4.3 Training Phase

The mean dataset will be derived from the original training dataset. As M files per speaker are used for training, the mean dataset will have M rows of values corresponding to each speaker, i.e., one row corresponding to each audio file of each speaker. The values in this row will be the mean of the values of that particular audio file. The procedure for calculation of these mean values has been described next. A Weighted mean vector is obtained from each audio sample in the training dataset. Let y_{ih} denote the hth MFCC feature of the ith frame of an audio sample. Then, obtain the histogram (with 20 bins) z_{ih} for all the y_{ih} values in a single column, where i varies from 1 to total number of frames in the audio sample that is 50 and h is 1 to 39. Let histogram be $p(z_{ih})$ for all values(z_{ih}). Then the weighted mean or the frame prototype for our experiment is given by the equation:

$$m_h = \sum_{i=1}^{50} z_{ih}\ p(z_{ih})\ , \qquad h = 1\ to\ 39 \qquad (4.2)$$

The weighted mean value for each column in the feature matrix is calculated and a mean vector corresponding to the audio sample is obtained of dimensions 1*39. Eg: suppose there are 43 speakers in all with M samples per speaker used for training. A weighted mean vector has to be obtained from each audio sample in the training dataset. The formula for obtaining the same has been discussed in detail above. The weighted mean value for each column in the feature matrix of the audio sample is calculated and a mean vector corresponding to the audio sample is obtained of dimensions 1*39. Since there are a total of 43 speakers and M audio samples per speaker are trained, the total size of training dataset after taking weighted mean=(43*M)*39. For further computations only these values will be required. We will be referring to this dataset as the ‘mean dataset’ throughout this work. Note that significant reduction in size of the training database to be used is achieved in the process as size of ‘mean dataset’ is significantly lesser than the size of the ‘original training dataset’ and only the ‘mean dataset’ needs to be stored and will be used by the proposed fuzzy system.

4.4.4 Testing Phase

MFCC features are to be extracted for each audio sample to be tested.

Now that we have the mean dataset and the test dataset, we are ready for testing. In order to recognize using fuzzy logic, Gaussian fuzzy membership function is selected.

$$\mu = e^{-(b-c)^2 / 2v^2} \qquad (4.3)$$

where c and v represent the mean and variance respectively. v is taken to be 1/√2.

Now, testing is done one audio sample at a time. For each audio sample, testing is done on a frame by frame basis.

a) First of all, the degree of membership of each frame in the test sample to each frame in the mean database (each frame in the mean database is a fuzzy prototype representing a sample) needs to be computed. Let the sample to be tested be x and the mean dataset obtained by equation 4.1 when computed for all training samples is $\{m_{kh}\}$ where h varies from 1 to 39 and k=No. of speakers*No. of samples per speaker . Now size of x=50*39 and size of m_h is (43*M)*39. Next, each frame j in the test sample is compared with each row k in the training dataset(m_h) and a degree of membership of frame j to k is obtained, denoted as under:

$$\mu_{jk} = e^{-||x_{jh} - m_{kh}||^2} \qquad (4.4)$$

Where j varies from 1 to total no. of frames in the test sample, in this case, 1 to 50.
k varies from 1 to total no. of values in the mean dataset, in this case, 1 to (43*M).
h varies from 1 to 39, as both test sample and mean values are 39 dimensional.

b) Next step is calculation of degree of membership of each frame in the test sample to each speaker. Now, in the resultant matrix of membership degrees obtained, first 9 values are membership degrees corresponding to user1, next 9 corresponding to speaker 2 and so on. So we take an average of first 9 values to represent degree of membership to speaker1(or class

1), average of next nine values to represent degree of membership to speaker2 and so on. μ_{ij} represents membership of frame j in the test sample to speaker p.

$$\mu_{jp} = \frac{\left[\sum_{k=(p-1)*M+1}^{(p-1)*M+M} \mu_{jk}\right]}{M} \qquad (4.5)$$

j varies from 1 to total number of frames in test sample, which is 50 here.
p varies from 1 to total number of speakers, which is 43.
k varies from 1 to total no. of values in the mean dataset, which is (43*M) in this case.

Now according to the explanation above, μ_{j1} for example, will be given by,

$$\mu_{j1} = \frac{\left[\sum_{k=1}^{9} \mu_{jk}\right]}{9}$$

And it will represent the membership degree of frame j to user1.

c) Finally, assign each frame in the test sample to the class to which its corresponding membership degree is the maximum.
Out of all the 43 membership degree values (corresponding to speakers 1 to 43) per frame, the frame is assigned to the class to which its membership degree value is maximum. Eg : If for a frame, membership degree value corresponding to class 13=1, then that frame will be classified to class 13.

$$class_j = p, \text{ where } \mu_{jp} \text{ is maximum} \qquad (4.6)$$

j varies from 1 to total number of frames in the test sample, which is 50.

and p varies from 1 to total number of speakers, which is 43 here.

d) Finally, at the end, every frame in the test sample has a corresponding class no. assigned to it. Since in this case there are a total of 43 speakers, every frame in the test sample will have a corresponding class no. that is between 1 and 43. The test sample will be classified to the class that is most frequent amongst all its frames.

e) This procedure is repeated for all the 43 test audio samples.

4.4.5 Error Computation

System Error is represented in terms of percent misclassifications.

$$Percent\ misclassifications = \frac{No.of\ samples\ classified\ incorrectly}{Total\ no.of\ samples} \quad (4.7)$$

The goal is that percent misclassifications have to be minimized, and the system efficiency has to be improved. The formula for percent efficiency of the system is as under:

$$Percent\ efficiency = 100 - Percent\ misclassifications \quad (4.8)$$

CHAPTER 5

EXPERIMENTAL SETUP

5.1 The VidTIMIT Database

The speaker identification experiments have been conducted on audio samples taken from the VidTIMIT audio-video dataset. We first describe the VidTIMIT database [42] that is used in this research. The VidTIMIT database is an audio-visual database containing recordings of 43 people reciting sentences from TIMIT corpus [43]. It has been recorded in 3 sessions with a gap of 7 days between sessions 1 and 2 and 6 days between sessions 2 and 3.The gap between sessions accounts for the possibility of mood and appearance changes that may occur in real life. There are a total of 10 sentences per person, 6 of them recorded in session 1 and two each in sessions 2 and 3. Two sentences are common to all speakers while the other eight sentences are different for each speaker. For our research, experiments have been carried out on all the 43 speakers in the Vid-TIMIT dataset. Out of the ten audio samples per speaker, nine have been used for training and one audio sample per speaker has been used for testing.

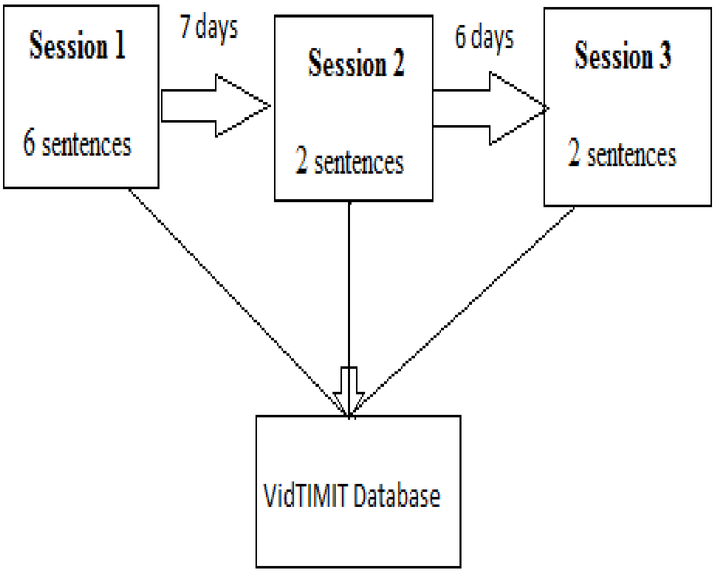


Figure 5.1: Audio recording in the VidTIMIT Database

5.2 Architectural Design of the proposed system

A block diagram giving the overview of the architectural design of the proposed system has been shown in Fig 5.2 given on the next page. First, acquisition of a voice sample takes place. After which, preprocessing and enhancement takes place in order to convert the query voice sample into a good quality voice sample. The MFCC's are extracted from the good quality voice sample and the extracted feature set will be what we will work on. The extracted feature set presented to the system will be compared against templates stored in the database to find a match. The stored templates in the database are nothing but the previously acquired samples from each speaker, from which MFCC features have been already extracted and stored in the database. After matching, the fuzzy classifier will assign a class to the presented sample and will classify it as belonging to a specific subject/speaker.

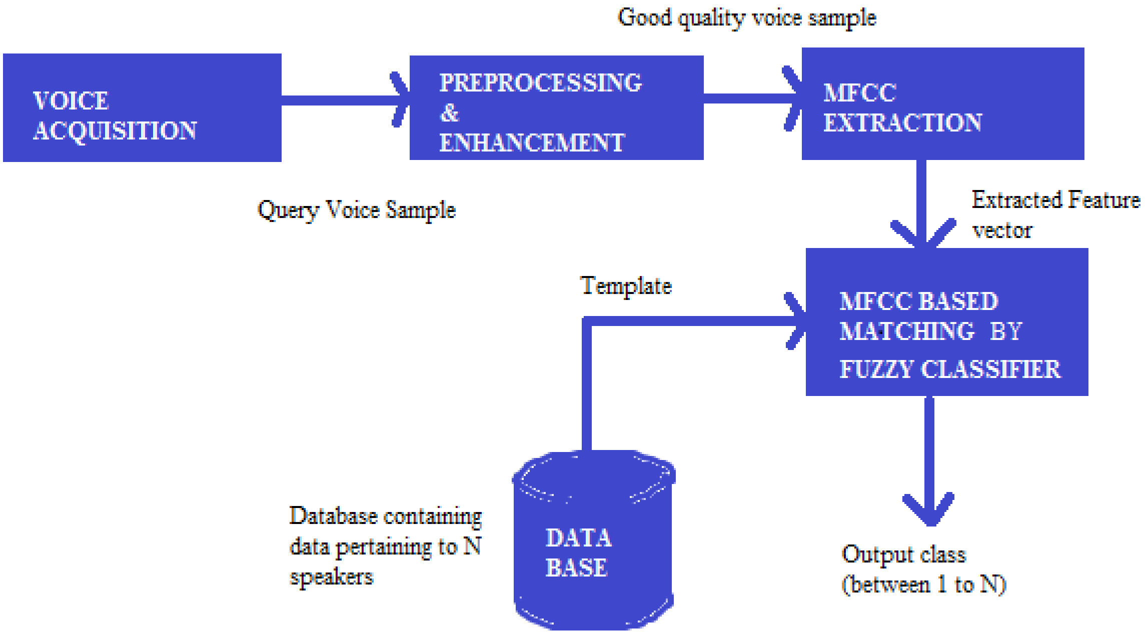

Good quality voice sample
VOICE ACQUISITION
PREPROCESSING & ENHANCEMENT
MFCC EXTRACTION
Query Voice Sample
Extracted Feature vector
Template
MFCC BASED MATCHING BY FUZZY CLASSIFIER
Database containing data pertaining to N speakers
DATA BASE
Output class (between 1 to N)

Figure 5.2 : The proposed system block diagram

5.3 Preprocessing and Enhancement

To compensate for any inconsistencies obtained in the audio samples, preprocessing steps are needed. For feature extraction and preprocessing from each audio sample, we use the short-term analysis technique using a 25ms window with 50% overlap between adjacent windows. Before extraction of MFCC features, the following preprocessing steps are applied:

a) pre-emphasis is carried out to reduce the high frequency falloff, and improve the signal to noise ratio.
b) hamming window is applied to each segment to minimize spectral leakage.

Both of the above steps have been explained earlier in detail in chapter 2.

Only after preprocessing, features will be extracted from the voice samples and used for research.

5.4 MFCC Feature Extraction

As discussed in the previous chapters, the Mel-frequency cepstral coefficients (MFCC) features give the most superior performance. MFCC features have to be extracted from all the audio samples of each speaker. Each frame is converted to 12 MFCCs plus a normalized energy parameter. The first and second derivatives (D's and DD's) of MFCCs and energy are estimated, resulting in 39 numbers representing each frame. Detailed description about this has been given earlier in chapter 2. All the frames in the sample collectively constitute the feature matrix of MFCC features for the sample, as shown in Fig 5.3 and based on the lines of work already done earlier [44]. The above-mentioned steps are the most widely used and form a part of most (if not all) speech and speaker feature extraction systems

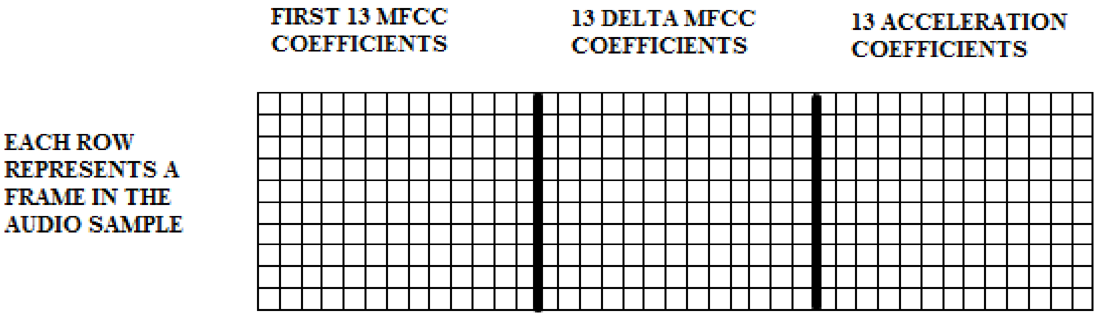

FIRST 13 MFCC COEFFICIENTS
13 DELTA MFCC COEFFICIENTS
13 ACCELERATION COEFFICIENTS
EACH ROW REPRESENTS A FRAME IN THE AUDIO SAMPLE

Figure 5.3: Feature matrix of MFCC features for a single audio sample.

5.5 Implementation Details for various classifiers

All the preprocessing and feature extraction steps described in the earlier sections of this chapter have to be carried out for all the training as well as test samples. The proposed fuzzy classifier will then apply the matching step on the extracted features. Also, some of the other commonly used techniques like Neural Network classifier, Nearest Neighbour classifier, Hidden Markov Models and Gaussian mixture clustering, will also be used for matching. Then the results of all these methods will be compared.

Speaker identification is carried out using the proposed fuzzy nearest neighbour classifier as per the procedure described in Chapter 4 and given by equations 4.2 to 4.8. We will be using nine audio samples per speaker for training and one audio sample per speaker for testing. This means that in equation 4.5, M=9. Each audio sample contains 50 frames. With M=9, as explained in Section 4.4.2, the size of the original training dataset=19350*39. The total size of the mean dataset= 387*39 as per the explanation given in Section 4.4.3. It is worth noting that size has been reduced from original 19350*39 to only 387*39. As explained in section 4.4.4, since there is only audio sample per speaker that is used for testing, size of test dataset=(43*1*50)*39= 2150*39.

Speaker identification using the Neural Network Classifier has to be done as per the details given in Chapter 3. There is an input layer, hidden layer and output layer. There are 100 neurons in the hidden layer. In the past, similar work has been done by J. Oglesby and J . S. Mason , titled ‘Speaker Recognition with a Neural Classifier’[45].

Speaker identification using the k-Nearest Neighbour classifier is performed as explained in Chapter 3 and making use of the Euclidean distance metric and the value of k=1. The value of k=1 has been determined experimentally. K-Nearest Neighbour classifier has been used for speech/speaker recognition experiments in the past as well [46].

For, HMM classification, the no. of states= 3 and the no. of mixtures=2. The details regarding HMM’s have already been discussed in Chapter 3. It is a very popular method for speaker recognition and a number of papers have been published in this past related to the use of Hidden Markov Models in speech/speaker recognition like ‘Thai Connected Digit Speech

Recognition Using Hidden Markov Models'[47], 'Evaluation and Modification of Cepstral Moment Normalization for Speech Recognition in Additive Babble Ensemble'[48], etc.

In the case of Gaussian mixture clustering, first cluster modelling has to be carried out and once the model has been designed, the clustering is carried out. The details about this have been discussed earlier in chapter 3. A variant of Gaussian mixture clustering, the Gaussian mixture models are one of the most commonly used techniques in speaker recognition and have been used by many researchers in the past including G. Suvarna Kumar et. al. in 'Speaker Recognition Using GMM' [49]

The efficiency in percent for each of these techniques, along with the time taken for classification has to be noted. This has been done at different points like for: 5 speakers, 10 speakers, 15 speakers, 20 speakers, 25 speakers, 30 speakers, 35 speakers, 40 speakers and finally for 45 speakers and the results obtained at each step are noted. Then comparative graph of all the techniques is plotted that depicts graphically the results obtained at each step.

CHAPTER 6

RESULTS AND DISCUSSION

6.1 Results of the proposed Fuzzy Classifier

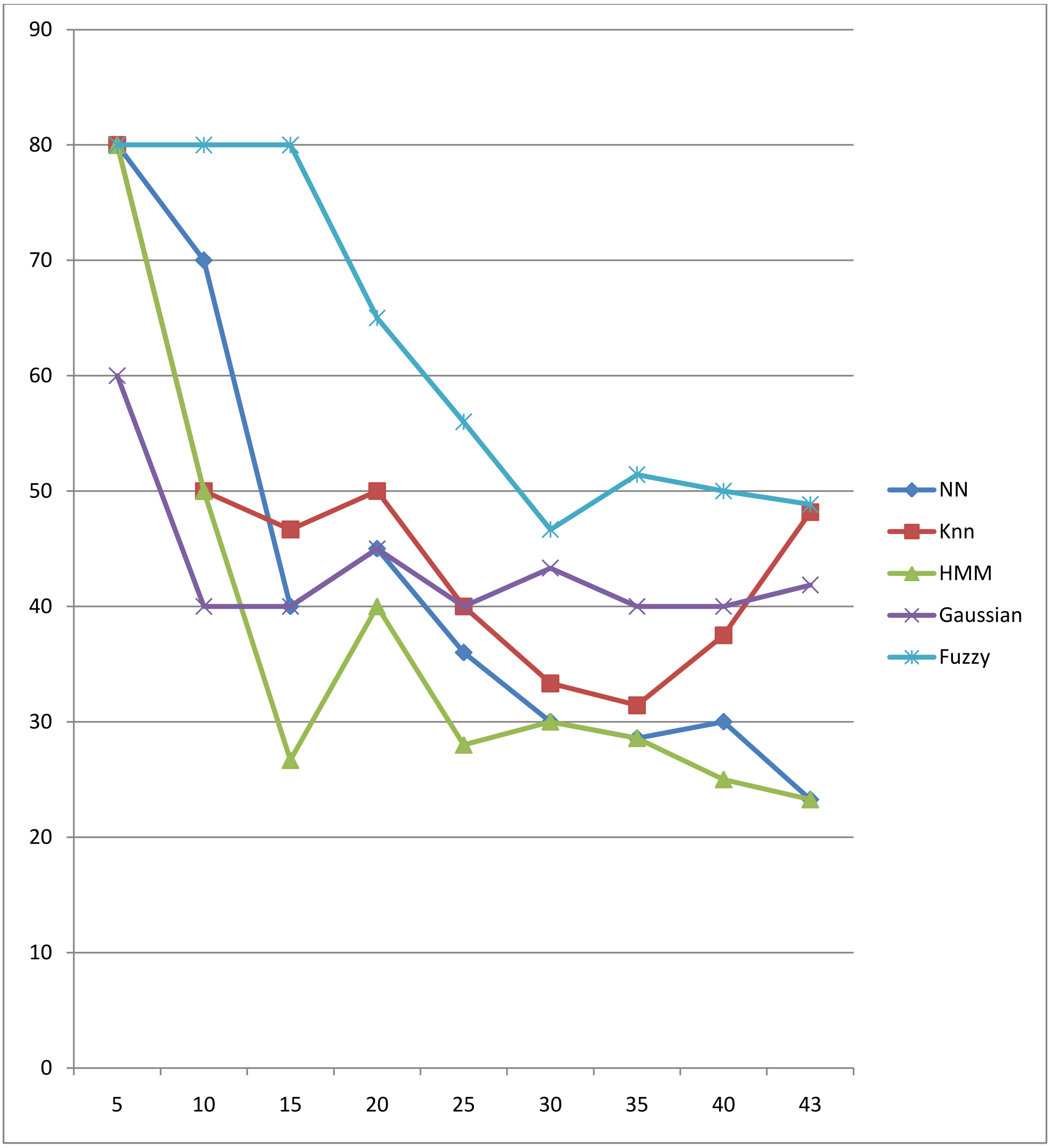
The speaker identification experiments have been carried out using MATLAB 7.9.0 on an Intel® Core ™ i5 CPU 2.4 GHz on samples taken from the Vid-TIMIT database The working of the proposed classifier has already been explained in Chapter 4 in equations 4.2 to 4.8. Note that out of the ten audio samples per speaker, nine have been used for training and one for testing. So, the value of M in equation 4.5 is equal to 9. The results obtained are listed in the last column of Table 6.1 and indicate a high level of accuracy. It can be seen that till 15 speakers, we get a very high level of accuracy that is 80% by the proposed fuzzy nearest neighbour classifier. After, increasing the number of speakers beyond this point, performance starts to gradually deteriorate. However, we still manage to get an accuracy of about 50% with 43 speakers. A detailed comparison of the obtained results with the results obtained by other methods has been given in the following section.

6.2 Comparison with other Methods

The experimental results obtained after testing from various techniques: The Nearest Neighbour classifier, the Neural Network classifier, the Hidden Markov Model classifier, Gaussian Mixture Clustering and the proposed Fuzzy Nearest Neighbour classifier for subsets of 5 speakers, 10 speakers, 15 speakers, 20 speakers, 25 speakers, 30 speakers, 35 speakers, 40 speakers and 43 speakers were noted. This was done to study the effect on accuracy as the number of speakers was increased. The computation time was also noted. The operational details about all the techniques have been discussed in the previous chapters. The obtained results have been depicted in graphical as well as tabular form and are as shown in Figure no. 6.1 and Table no. 6.1 respectively. As can be seen from the figure as well as the table, the proposed fuzzy classifier performs relatively better than all the other previously used techniques at all points. When we analyze the results obtained in detail, we find out that in the beginning when the system is small and there are only 5 speakers, we get an efficiency of 80% with all the techniques except Gaussian mixture clustering which gives an efficiency of 60%. As, this is a difficult dataset we have worked on, it is a reasonable performance. We may say that till this stage all the techniques give good performance. For 10 speakers, the proposed fuzzy classifier gives the best results, i.e. it still gives an efficiency of 80%, followed by the neural network classifier which gives an efficiency of 70%. The rest of the techniques give an efficiency of 40-50%. On increasing the no. of speakers to 15, the proposed fuzzy classifier still gives an efficiency of 80%, which is remarkable but the other classifiers give efficiency in the range of 25-40%. As we keep on increasing the no. of speakers beyond this point to 20, 25, 30, 35, 40 and 43 we find that the only technique that gives us reasonable results at all points is the proposed fuzzy classifier, whose efficiency never falls below 45% at any point. So, without doubt, the proposed fuzzy classifier is the best technique out of all. It may also be noted that though the Gaussian mixture clustering did not give notable results at the early stages, with lesser number of speakers, however it does not suffer from scalability issues & manages to give near reasonable performance at all points. The performance never falls below 40% at any point. We may say that after the proposed fuzzy nearest neighbour classifier, Gaussian mixture clustering is the most reliable of all, as it manages to give a near reasonable performance throughout. Hidden Markov Model and Neural Network classifier, on the other hand, are the least reliable as their performance drops significantly as we keep on increasing the number of speakers. Finally when we see the results obtained for 43 users, we find that the proposed Fuzzy Nearest Neighbour Classifier is the most efficient with an efficiency of 48.84%, followed by Nearest Neighbour classifier with an efficiency of 48.17%, followed by the Gaussian mixture clustering with an efficiency of 41.86%. It is also worth noting that the proposed fuzzy classifier gives the best results at each point, which leads us to believe that the proposed fuzzy classifier comprehensively outperforms all the other techniques that have been used for comparison. The efficiency is around 50% at all points using a behavioural biometric and that too for such a difficult dataset. So, the proposed fuzzy classifier has the scope to be used for real-time speaker identification experiments.

No. of speakers	Neural Network classifier	Nearest Neighbour Classifier	Hidden Markov Model	Gaussian clustering	Proposed Fuzzy Nearest Neighbour Classifier
5	80%	80%	80%	60%	80%
10	70%	50%	50%	40%	80%
15	40%	46.67%	26.67%	40%	80%
20	45%	50%	40%	45%	65%
25	36%	40%	28%	40%	56%
30	30%	33.33%	30%	43.33%	46.67%
35	28.57%	31.43%	28.57%	40%	51.43%
40	30%	37.5%	25%	40%	50%
43	23.26%	48.17%	23.26%	41.86%	48.84%

Table 6.1: Results obtained after matching by the different techniques: Nearest Neighbour, Neural Network, HMM, Proposed Fuzzy Classifier and Gaussian Mixture Clustering



Xaxis= No. of speakers ,

Y axis= Accuracy in percent

Figure 6.1 : Results obtained after classification by the different classifiers: Nearest Neighbour(k-NN with k=1), Neural Network(NN), HMM, Proposed Fuzzy Nearest Neighbour Classifier and Gaussian Mixture Clustering

6.3 Space and Time Complexity comparison of different algorithms

All the other techniques mentioned use the ‘original training dataset’ and require it to be stored. On the other hand, the proposed fuzzy classifier does on operate on the original training dataset. Rather, it uses the ‘mean dataset’ and requires only this dataset to be stored in the computer’s memory. It only uses the ‘original training dataset’ to compute the ‘mean dataset’. After that the ‘original training dataset’ may be discarded as only the ‘mean dataset’ will be used for further computation. As the ‘mean dataset’ is much smaller in size as compared to the ‘original training dataset’, the space complexity of the proposed fuzzy classifier algorithm is much less than all other algorithms.

As far as the time complexity is concerned, the proposed fuzzy nearest neighbour classifier takes the least computation time and the neural network classifier takes the maximum amount of time. All the techniques and their corresponding computation time along with the size of training dataset, efficiency in percent, for 43 users are listed in table 6.2, given on next page. Time taken by different techniques at different points has also been depicted in the form of a table, table 6.3. It is worth noting that except for points, 5 and 10 speakers where Nearest Neighbour classifier is faster than the proposed fuzzy classifier, the proposed fuzzy Nearest Neighbour classifier is the fastest at all other points. We may say that the proposed fuzzy Nearest Neighbour classifier has a time complexity that is comparable to Nearest Neighbour Classifier. So, we may conclude that taking both space and time complexity into consideration, the proposed fuzzy classifier, is better than all the other techniques by a reasonable margin.

CLASSIFIER	TRAINING VECTOR SIZE	EFFECIENCY	COMPUTATION TIME
• Fuzzy Nearest Neighbour	387*39	48.84%	7.7 seconds .
• Neural Network(100 neurons)	19350*39	23.26%	1686.48 seconds
• HMM	19350*39	23.26%	1420.71 seconds
• Gaussian mixture clustering	19350*39	41.86%	254.57 seconds
• Nearest Neighbour classifier	19350*39	41.87%	23.12 seconds

Table 6.2. : Comparative analysis of the different matching techniques on parameters: size of training dataset, efficiency in percent and computation time for 43 users.

No. of speakers	Neural Network	Nearest Neighbour	HMM	Gaussian clustering	Proposed Fuzzy Nearest Neighbour Classifier
5	93.78 sec	0.34 sec	58.18 sec	1.08 sec	1.93 sec
10	337.85 sec	1.35 sec	137.26 sec	8.68 sec	1.96 sec
15	683.30 sec	3.08 sec	365.09 sec	30.83 sec	2.02 sec
20	536.95 sec	5.62 sec	402.14 sec	30.13 sec	2.45 sec
25	1122. 24 sec	7.96 sec	541.75 sec	33.41 sec	3.06 sec
30	1021.68 sec	11.00 sec	995.43 sec	127.85 sec	3.86 sec
35	1172.92 sec	15.11 sec	1215.76 sec	170.67 sec	5.14 sec
40	2048.08 sec	19.55 sec	1386.35 sec	219.198 sec	6.41 sec
43	1686.48 sec	23.12 sec	1420.71 sec	254.57 sec	7.7 sec

Table 6.3: Time in seconds, taken by the different techniques: Nearest Neighbour, Neural Network, HMM, Proposed Fuzzy Nearest Neighbour Classifier and Gaussian Mixture Clustering for matching at different points

---

6.4 Performance of other techniques with the 'mean dataset'

Later, for the purpose of comparison, all the other previously used techniques (except the proposed method), that have been used for comparison with the proposed nearest neighbour fuzzy classifier, were also trained using the 'mean dataset' instead of the 'original training dataset' and the results obtained with the 'mean dataset' noted. They are as shown in table 6.4 given on the next page. As can be seen by comparing the two tables, Table 6.2 & Table 6.4 , all the other previously used techniques give better results with the 'original training dataset'. Time taken is lesser in case of 'mean dataset' used for training, but this is at the expense of a large decrease in efficiency. So, we did not alter these methods to work with the 'mean dataset' and used the other matching techniques along with the 'original training dataset' itself.

CLASSIFIER	TRAINING VECTOR SIZE	EFFECIENCY	COMPUTATION TIME
• Neural Network(100 neurons)	387*39	9.3%	528.19 seconds
• Hidden Markov Model	387*39	-	-
• Gaussian mixture clustering	387*39	16.28%	2.47 seconds
• Nearest Neighbour classifier	387*39	32.56%	2.19 seconds

Table 6.4: Results obtained by using the previously used matching techniques along with the 'mean dataset' used for training in place of the 'original training dataset' for 43 users

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

A new classifier for speaker identification that makes use of MFCC features and their corresponding delta and acceleration values has been proposed in this thesis. The novelty lies in the use of a new classification technique based on fuzzy logic. The testing of our approach was done on phrases of the Vid-TIMIT database. Comparisons were made by testing the same data using Neural Network Classifier, Nearest Neighbour Classifier, Hidden Markov Model and Gaussian Mixture Clustering. Based on our observations and obtained results, the following conclusions can be made:

- The efficiency achieved by the proposed fuzzy nearest neighbour classifier is more than all the other techniques that are commonly used for speaker identification.
- The space complexity of the proposed algorithm is lesser than the space complexity of the existing technology for speaker recognition.
- The computation time taken by the proposed fuzzy nearest neighbour classifier is comparable to the computation time taken by the nearest neighbour classifier and is much lesser than all other techniques.
- The proposed fuzzy classifier has a classification time suitable for use in automated speaker recognition systems.
- The low complexity of the design and the low cost of implementation of the system make this technique a very feasible one for practical purposes.
- The performance of the proposed system is least affected by scalability, which is a very common problem with voice based biometric systems.

7.2 Future Work

One of the biggest issues till date hindering the practical viability of voice based biometric systems is their unproven scalability. In this regard, the results obtained by the proposed fuzzy classifier till 15 speakers were found to be excellent and much better as compared to all the other techniques commonly in use. However, on increasing the number of speakers, beyond this point, the results were still better than all the other techniques that were used for comparison, but they were not exceptionally good. So, one of the biggest challenges for the future remains to improve the performance of the system as to achieve exceptionally good results as compared to the other techniques at all points. Another goal is to expand the capability of the system so that the system performs genuine and impostor user distinction. This means that the system would contain training samples pertaining to some users, who will be called the “genuine users”, while all others would be “impostors” and when an impostor user sample will be presented to the system, the system would explicitly state so. By setting some threshold we can do this task. If a presented sample’s matching score is less than the threshold value for each of the training samples, then the system would not classify the presented sample into any of the classes and would term it to be an impostor sample. If this functionality is added to the proposed system, it would become much more relevant for practical implementation on a wider scale.

REFERENCES

[1] http://www.biometrics.gov/documents/biohistory.pdf

[2] Karthik Nandakumar,"INTEGRATION OF MULTIPLE CUES IN BIOMETRIC SYSTEMS", *a thesis Submitted to Michigan State University*.

[3] Sharat S. Chikkerur," ONLINE FINGERPRINT VERIFICATION SYSTEM", *A thesis submitted to the Faculty of the Graduate School of the State University of New York at Buffalo.*

[4]http://books.google.co.in/books?id=GnwyMm0QH5UC&pg=PA658&lpg=PA658&dq=a+typical+biometric+system+consists+of+4+modules&source=bl&ots=PTdkQpeScu&sig=swlp1UqXOfQL-WWP

[5]http://www.unbankedtrends.com/index.php/2010/07/biometrics-verification-vs-identification/

[6] http://www.webopedia.com/TERM/P/physical_biometric.html

[7] http://www.webopedia.com/TERM/B/behavioral_biometric.html

[8] http://www.authentify.com/solutions/voice_biometrics.html

[9] http://www.biometrics.gov/documents/biohistory.pdf

[10] http://en.wikipedia.org/wiki/Speaker_recognition

[11] Zhiping Dan,Sheng Zheng, Shuifa Sun, Ren Dong," Speaker Recognition based on LS-SVM**",** *The 3rd International Conference on Innovative Computing Information and Control (ICICIC'08).*

[12] Nima Yousefian, Azarakhsh Jalalvand, Pooyan Ahmadi, Morteza Analoui," Speech Recognition with a Competitive Probabilistic Radial Basis Neural Network*", 2008 4th International IEEE Conference "Intelligent Systems"*

[13] Hemant A. Patil, Prakhar Kant Jain, Robin Jain,” A Novel Approach To Identification Of Speakers From Their Hum”**,** *2009 Seventh International Conference on Advances in Pattern Recognition*

[14] Z. Uzdy,”Human Speaker Recognition Performance of LPC Voice Processors**”,** *IEEE Transactions on Acoustics, Speech, and Signal processing*, vol. assp-33, no. 3**,** june 1985

[15] Rajparthiban Kumar, Aravind CV, Kanendra Naidu, Anis Fariza,” Development of a Novel Voice Verification System using Wavelets”, *Proceedings of the International Conference on Computer and Communication Engineering 2008 May 13-15*, 2008 Kuala Lumpur, Malaysia

[16] Dr. Gwyn P. Edwards,” A Speech/Speaker Recognition and Response System”, *Acoustics, Speech and Signal Processing, IEEE International Conference on ICASSP,1980*

[17] Wu Guo, Yanhua Long, Yijie Li, Lei Pan, Eryu Wang, Lirong Dai*, ”* iFLY System For The NIST 2008 Speaker Recognition Evaluation”, *ICASSP '09 Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing.*

[18] Ramon F. Astudillo, Dorothea Kolossa, Reinhold Orglmeister,” Uncertainty Propagation for Speech Recognition using RASTA Features in Highly Nonstationary Noisy Environments”, *ITG-Fachtagung Sprachkommunikation 8- 10 October 2008 in Aachen*

[19] Tilo Schiirer,” An Experimental Comparison Of Different Feature Extraction And Classification Methods For Telephone Speech”, *2nd IEEE Workshop on Interactive Voice Technology for Telecommunications Applications* **(***IvTTA94)*

[20] Ross, A. and Jain, A. K., “Multimodal biometrics:an overview,” *Proc. EUSIPCO*, pp. 1221-1224, Sept.2004.

[21] D.Reynolds, T.Quatieri, and R.Dunn, "”Speaker Verification Using Adapted Mixture Models”, , *DigitalSignal processing, 200010,* pp.181-202.

[22] Zhiping Dan.,Sheng Zheng.,Shuifa Sun.,Ren Dong., "Speaker Recognition Based on LS-SVM" ,*The 3rd International Conference On Innovative Computing And Information And Control,2008.*

[23] wwwold.ece.utep.edu/research/webfuzzy/docs/paper_3_fuzz96.doc

[24] Y. Arriola, R A Carrasco," Integration Of Multilayer Perceptron And Markov Models For Automatic Speech Recognition", *UK IT 1990 Conference*

[25] http://citeseerx.ist.psu.edu/viewdoc/summary?doi=?doi=10.1.1.126.4298, Tongtao Zheng, Dat Tran and Michael Wagner" Fuzzy Nearest Prototype Classifier Applied to Speaker Identification"

[26] Jyh-Shing Roger Jang and Jiuann Jye-Chen," Neuro-Fuzzy and Soft Computing for Speaker Recognition**",** *Proceedings of IEEE International Conference on Fuzzy Systems,* PP. 663-668,Barcelona Jul 1997.

[27] http://en.wikipedia.org/wiki/Cepstrum

[28] http://www.cic.unb.br/~lamar/te073/Aulas/mfcc.pdf , "The MFCC", Aldebaro Klautau

[29] http://www.mathworks.in/help/toolbox/signal/ref/hamming.html

[30] http://en.wikipedia.org/wiki/Mel_scale

[31] http://en.wikipedia.org/wiki/Mel-frequency_cepstrum

[32] http://uhaweb.hartford.edu/compsci/neural-networks-history.html

[33] Mark Hudson Beale, Martin T. Hagan and Howard B. Demuth, **Neural Network Toolbox 7.0, User's Guide**, Copyright 1992-2010 by the Math Works Inc.

[34] http://www.scholarpedia.org/article/K-nearest_neighbor

[35] http://en.wikipedia.org/wiki/K-nearest_neighbor_algorithm

[36] http://www.mathworks.in/help/toolbox/bioinfo/ref/knnclassify.html

[37] http://www.mathworks.in/help/toolbox/stats/bq_679x-24.html#bra9fvn

[38] http://www.mathworks.in/help/toolbox/stats/brklrj3.html#brklr93-1

[39] M. Hanmandlu, J. Grover, V. K. Madasu, S. Vasikarla, "Input Fuzzy Modeling for the Recognition of Handwritten Hindi Numerals", *International Conference on Information Technology (ITNG'07).*

[40] **Fuzzy Modelling and Control** by Andrzeg Piegat.

[41] http://www.scholarpedia.org/article/Fuzzy_classifiers

[42] Sanderson, C., Biometric person recognition : face,speech, and fusion. *VDM Verlag, June 2008.*

[43] Garofolo, J. S., Lamel, L.F., Fisher, W. M., Fiscus, J. G., Pallett, D. S., and Dahlgren, N. L., "The DARPA TIMIT acoustic-phonetic continuous speech corpus CDROM," *NIST order number PB91-100354, 1992.*

[44] Dhaval Shah, Kyu J. Han, Shrikanth S. Nayaranan,"A Low-Complexity Dynamic Face-Voice Feature Fusion Approach to Multimodal Person Recognition", *2009 11th IEEE International Symposium on Multimedia.*

[45] J. Oglesby and J. S. Mason, "Speaker Recognition with a Neural Classifier" *University College, SWANSEA, UK.*

[46] Golipour l., O' Shaughnessy D. ,"Context-independent phoneme recognition using a K-Nearest Neighbour classification approach", *ICASSP 2009, IEEE International Conference on Acoustics, Speech and Signal Processing,* Pages 1341-1344.

[47] Amarin Deemagarn, Asanee Kawtrakul," Thai Connected Digit Speech Recognition Using Hidden Markov Models", *SPECOM'2004: 9th Conference Speech and Computer,*St. Petersburg, Russia, September 20-22, 2004

[48] Roberto Togneri, Aik Ming Toh, Sven Nordholm, "Evaluation and Modification of Cepstral Moment Normalization for Speech Recognition in Additibe Babble Ensemble", *Proceedings of the 11th Australian International Conference on Speech Science & Technology,* Pages 94-99

[49] G.Suvarna Kumar, K.A.Prasad Raju, Dr.Mohan Rao CPVNJ, P.Satheesh, "Speaker Recognition Using GMM", *International Journal of Engineering Science and Technology Vol. 2(6), 2010, 2428-2436.*